Universidad de las Ciencias Informáticas,

Facultad 1

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título: Indexación de archivos del sistema para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0

Autor:

Yoandri Pablo Martínez Magaña

Tutor(es):

Ing. Gladys M. Peñalver Romero

Ing. Dairelys García Rivas

La Habana, Junio de 2015

"Año 57 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro ser autor de este trabajo y autorizo a la Facultad 1 de la Universidad de Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste se firma la presente declaración jurada de autoría en La Habana a los ____ días del mes de _____ del año _____.

_____

Yoandri Pablo Martínez Magaña

Autor

_____ _____

Ing. Gladys M. Peñalver Romero Ing. Dairelys García Rivas

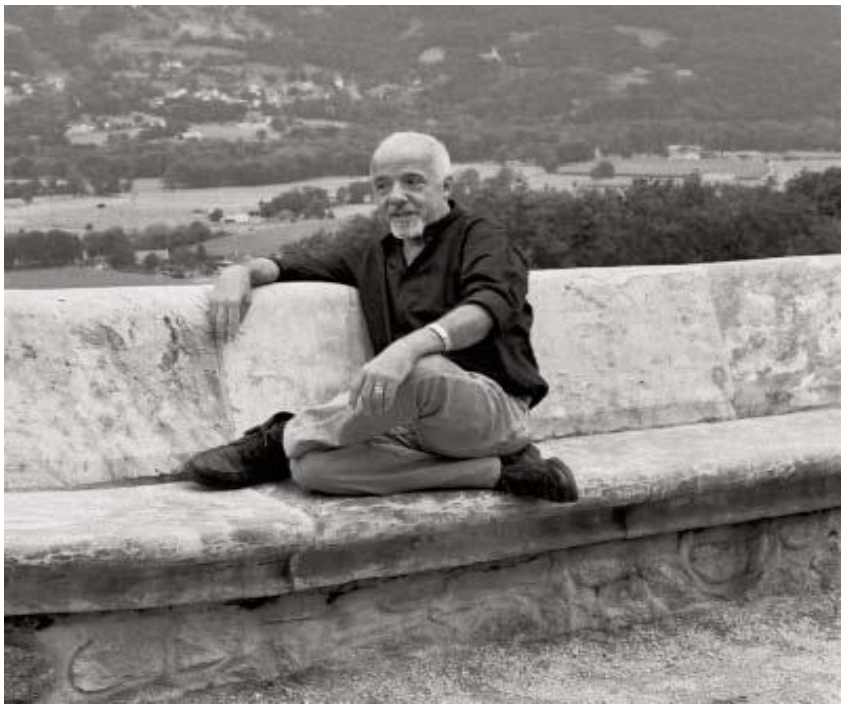
Tutoras

Datos de contacto

Autor: Yoandri Pablo Martínez Magaña.
Correo: ypmagana@estudiantes.uci.cu
Universidad de las Ciencias Informáticas, La Habana, Cuba.

Tutora: Ing. Gladys Marsi Peñalver Romero.
Correo: gmpenalver@uci.cu
Universidad de las Ciencias Informáticas, La Habana, Cuba.

Tutora: Ing. Dairelys García Rivas.
Correo: dgrivas@uci.cu
Universidad de las Ciencias Informáticas, La Habana, Cuba.



"El mundo está en las manos de aquellos que tienen el coraje de soñar y correr el riesgo de vivir sus sueños."

Paulo Coelho

AGRADECIMIENTOS

Agradecerle primeramente a mi papá por todo el esfuerzo y apoyo brindado durante estos 5 años, no muchas veces te digo te quiero pero siempre ten presente que eres la persona más importante en mi vida.

A mis hermanas y mi madrastra por también estar presente en todo momento y brindarme su apoyo.

A mis segundos padres Cary y Osvaldo que siempre estuvieron pendientes a pesar de la distancia.

A mi primo Rafael por siempre estar presente en todo momento.

En fin a toda mi familia que de alguna manera u otra siempre han estado conmigo.

A mis tutoras por ser incondicionales y dedicarme parte de su tiempo, gracias por hacer que este sueño pudiera hacerse realidad; sin su ayuda hubiera sido imposible.

A Ubaldito por ser amigo y hermano de forma incondicional y siempre en los buenos y en los malos momentos darme su apoyo y consejos cuando me he encontrado en un punto sin salida.

A Ileana, gracias amor por siempre estar ahí para mi y aguantar toda mi malcriadez y a pesar de todo nunca haber cambiado tú forma ser.

A Ester, tita gracias por estar siempre ahí para mi y gracias por formar parte de mi vida.

A mis otros hermanos Raúl, Dayrol, Dennis, José y Daniel por todo el tiempo compartido juntos.

Todas aquellas personas, que de una manera u otra permitieron que este sueño se hiciera realidad.

DEDICATORIA

Este trabajo de diploma va dedicado a mi papá y en especial a mi mamá que a pesar de que por cosas de la vida no puede estar presente físicamente conmigo, siempre ha estado en cada momento de mi vida y de alguna manera me a ayudó a seguir a delante cuando he pensado que no tiene sentido seguir y sé que su mayor sueño era verme graduado, para ti mami este trabajo.

Yoandri Pablo Martínez Magaña.

RESUMEN

Las herramientas existentes para búsqueda de información en sistemas GNU/Linux tienen la limitante de que solo permiten realizar búsquedas en los metadatos de los ficheros, no así en el contenido de los mismos. Esto trae como consecuencia que se limite la capacidad de búsqueda, y además se ignoren posibles resultados que pudieran ser útiles al usuario. Por esta razón, el propósito del presente trabajo es desarrollar una herramienta para la búsqueda de contenido en archivos locales en la Distribución Cubana de GNU/Linux Nova Escritorio 5.0, mediante la indexación de archivos.

Para la implementación de la propuesta de solución se utiliza como lenguaje de programación Java, la metodología *OpenUp* para guiar el proceso de desarrollo y para su construcción las herramientas: Apache Solr *4.10.2, Tomcat* 7 y *NetBeans* 8. Obteniéndose como resultado de esta investigación una herramienta que facilita la búsqueda de información en archivos mediante búsquedas en su contenido, lo cual aporta mayor cantidad de resultados de búsquedas para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0.

Palabras claves: *archivos, búsqueda, indexación, información, recuperación.*

ÍNDICE

Índice

Índice

ÍNDICE DE FIGURAS

Índice

ÍNDICE DE TABLAS

Introducción

Introducción

En la actualidad, el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC's), unido al surgimiento y evolución de las redes de computadoras propicia una amplia difusión de información a nivel mundial. Desde los inicios de la actividad humana, la información constituye un baluarte en todo el proceso de comunicación. Desde las primeras civilizaciones se archivan, manipulan y se crean grandes volúmenes de la misma en bibliotecas. Pero en el presente su valor se incrementa, pues antes del surgimiento de la informática no existía la posibilidad de interrelacionarla y procesarla con rapidez.

La comunicación se extiende por todo el mundo, y el flujo de información es cada vez mayor. Detrás de todo este desarrollo tecnológico se encuentra la información como objeto de dicha revolución tecnológica. Esta pasa a tener una gran importancia, el flujo y la cantidad que se intercambia en formato digital crece de forma exponencial, demandando canales de comunicación rápidos y efectivos. Para el acceso a esa gran masa de datos se requieren soportes potentes; donde estos avances tecnológicos y de las telecomunicaciones se ponen a disposición de los usuarios.

La existencia de la información digital supera a la analógica. Las TIC's facilitan su manejo, gestión, análisis, almacenamiento y recuperación. Pero a su vez la organización, recuperación y búsqueda de dicha información se hace una tarea de gran dificultad, puesto que la misma se va a encontrar almacenada en diferentes localizaciones y existe una gran cantidad de la misma.

Los primeros algoritmos de búsqueda consistían en localizar datos almacenados en una pequeña base de datos local. Así surgieron los primeros sistemas de búsqueda, cuyo objetivo fundamental consistía en acelerar el proceso de búsqueda de cualquier información en una base de datos. Estos sistemas tienen la limitante de que solo son capaces de realizar búsquedas textuales y no en el contenido de los archivos. Por tanto surge el problema de organizar y catalogar dicha información de una mejor manera, donde se encuentre centralizada y organizarla según su tipo con el objetivo de poder localizarla en cualquier momento. Cuando crece el volumen de archivos almacenados se hacen necesarios mecanismos capaces de localizarlos, no solo por palabras claves que describen el título, el autor y última fecha de modificación sino que se abarquen otros metadatos como sería el contenido. A su vez estos presentan problemas a la hora

de responder a las necesidades y preferencias específicas de los clientes, ya que los resultados son inexactos, debido a que la mayoría de los buscadores se limitan a las características antes citadas y no al contenido del archivo.

Por lo tanto se hizo necesario introducir un nuevo concepto de búsqueda, que sea capaz de registrar la información de forma ordenada independientemente de que se solicite su búsqueda o no; por lo que surge la indexación. Con su aparición se aceleró considerablemente el proceso de búsqueda. La indexación no es más que un proceso de recuperación de información que tiene como objetivo principal satisfacer las necesidades de información de un usuario expresadas en lenguaje natural (Ochando, 2013). Como resultado le devuelve al usuario una lista de ficheros ordenados según el grado de relevancia de los mismos; esta se basa en la búsqueda continua en la *Web* o en una dirección local.

Muchas tecnologías de búsqueda e indexación de archivos existentes como son el motor de *Google*, *Bing* y *Ask,* que se utilizan para indexar y buscar contenidos no son válidas para el trabajo local; ya que su enfoque está dado para realizar búsquedas en la *Web*; esto imposibilita que se obtengan resultados de interés en la localización de la información en un equipo local.

Hoy existen entornos desconectados de la red y como consecuencia, grandes cantidades de información deben ser almacenadas en los equipos personales de cómputo. Los buscadores por defecto existentes para búsqueda de información en sistemas GNU/Linux, tienen la limitante de que solo permiten realizar búsquedas en las características principales de los archivos, como son: el nombre, el autor o fecha de modificación, y no así en el contenido de los mismos. Esto trae como consecuencia que se restrinja la capacidad de búsqueda; y además se ignoren posibles resultados que pudieran ser útiles al usuario. La Distribución Cubana de GNU/Linux Nova no está exenta de estos problemas, el buscador que trae el navegador de archivo Nautilus solo permite realizar una búsqueda básica basada en el nombre del archivo, no abarca otros parámetros como son: los metadatos de autor, propietario, tamaño y fecha de creación; todo esto incluyendo la principal limitación de los sistemas de búsquedas locales, que no profundizan en el contenido interno de los archivos por lo que se necesita una herramienta que sea capaz de realizar búsquedas en contenido mediante el uso de la indexación. A partir de lo analizado anteriormente se identificó el siguiente **problema de la investigación**: ¿Cómo facilitar el proceso de recuperación de

información de archivos locales para la búsqueda de datos en la Distribución Cubana de GNU/Linux Nova Escritorio 5.0?

Como resultado del análisis del problema, se define como **objeto de estudio** el proceso de recuperación de información de archivos locales mediante búsquedas en su contenido y como **campo de acción** la gestión de recuperación de información de archivos locales mediante búsquedas para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0.

Se determina como **objetivo general** desarrollar una herramienta para la búsqueda de contenido en archivos locales en la Distribución Cubana de GNU/Linux Nova Escritorio 5.0.

Para regir la investigación se plantea como **idea a defender**: con el desarrollo de una herramienta que contribuya a la recuperación de información de archivos locales mediante búsquedas en su contenido, para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0, se facilitarán las búsquedas realizadas por los usuarios.

Se definen los siguientes **objetivos específicos**:

- ✓ Elaborar el marco teórico de la investigación.
- ✓ Diseñar e implementar la solución propuesta.
- ✓ Probar el correcto funcionamiento de la solución.

Para dar cumplimiento a los objetivos específicos expuestos anteriormente se desarrollarán las siguientes **tareas de investigación**:

- ✓ Análisis de la bibliografía relacionada con la búsqueda de indexación en sistemas operativos.
- ✓ Definición de las tecnologías y herramientas a utilizar en el desarrollo del buscador para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0.
- ✓ Análisis y diseño de la solución propuesta.

- ✓ Implementación de las funcionalidades definidas en la solución propuesta.
- ✓ Realización de pruebas para verificar el correcto funcionamiento de la solución desarrollada.

Con la culminación de todas las tareas y actividades desarrolladas en el proceso de investigación se esperan los siguientes **resultados**:

- ✓ Una herramienta que permita la recuperación de información de archivos locales mediante búsquedas en su contenido, metadatos y facilita el resultado de la localización de la información realizadas por los usuarios.
- ✓ Documentación asociada a la aplicación desarrollada.

Para el desarrollo de la investigación se utiliza el **método teórico** Analítico-Sintético para el estudio de las bibliografías para realizar la fundamentación teórica de la investigación y sintetizar los conceptos fundamentales como búsqueda, indexación y metadatos que sean necesarios para la solución del problema; extraer los elementos más relacionados e importantes con el objeto de estudio.

El presente trabajo está dividido en 3 capítulos.

Capítulo 1: Fundamentación teórica. En este capítulo se analizan los conceptos relacionados al dominio de la investigación, lo cual posibilita una mejor comprensión del tema. Se aborda lo referente a algunas de las herramientas que utilizan la indexación para realizar las búsquedas, así como tecnologías a usar para llegar a la solución y la metodología de desarrollo de software.

Capítulo 2: Propuesta de Solución. En este capítulo se define la propuesta del sistema a través de sus características y los requisitos funcionales y no funcionales, así como todo lo referente a la arquitectura del mismo, el diseño de la solución y patrones de diseño a utilizar.

Capítulo 3: Implementación y Pruebas de la Solución Propuesta. En este capítulo se documenta la fase de implementación, así como los diagramas correspondientes para un mejor entendimiento y los estándares

de código a utilizar en el desarrollo de la misma. Además de determinar mediante las pruebas definidas la validez del sistema.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

La búsqueda de información constituye un proceso fundamental para la obtención de conocimiento. Por lo tanto conocer sus principales conceptos, tecnologías y herramientas existentes; así como el uso de la indexación de ficheros facilita mediante el uso de índices aumentar las búsquedas realizadas en el sistema y una mayor cantidad de información a analizar.

Para lograr un mejor entendimiento sobre la problemática a resolver, se muestra a continuación un conjunto de conceptos que constituyen la base teórica del tema que se trata en la presente investigación para un mejor entendimiento.

1.2 Conceptos asociados al dominio de la investigación

Archivo informático

Es un grupo de datos estructurados que son almacenados en algún medio y pueden ser usados por las aplicaciones. La forma en que una computadora organiza, da nombre, almacena y manipula los archivos se denomina sistema de archivos y suele depender del sistema operativo y del medio de almacenamiento (Alegsa, 2013).

Es un conjunto de bits almacenado en un dispositivo. Un archivo es identificado por un nombre y la descripción de la carpeta o directorio que lo contiene. Son los equivalentes digitales de los archivos en tarjetas, papel o microfichas del entorno de oficina tradicional. Los archivos informáticos facilitan una manera de organizar los recursos usados para almacenar permanentemente datos en un sistema informático (Luna, 2012).

En la presente investigación el autor considera que un archivo informático es un conjunto de datos binarios que van a representar la información física almacenada en el sistema.

Indexación

La indexación es el proceso por el cual el buscador va rastreando el sitio en cuestión y a su vez va incorporando a su base de datos el contenido de ese Identificador de Recursos Uniforme (URL); es decir, se registra ordenadamente información para elaborar un índice (Ochando, 2013).

Técnica para recuperar los datos contenidos en un fichero o en una zona de memoria por medio de un índice que guarda la posición de los datos (thefreedictionary, 2009).

En la presente investigación el autor considera que indexación es proceso realizado para recuperar rápidamente del contenido de un fichero almacenado en memoria.

Recuperación de información

El conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución del problema planteado. En estas tareas desempeñan un papel fundamental los lenguajes documentales, las técnicas de resumen, la descripción del objeto documental (Tolosa & Bordignon, 2005).

La aplicación del conjunto de técnicas, métodos y actividades para buscar, localizar y recuperar de una manera eficiente en los diversos Sistemas de Recuperación de Información (SRI) la información relevante que requiere el usuario, y satisfacer así su necesidad de información (Oliván & Avilés, 2006).
En la presente investigación el autor considera que la recuperación de información es el conjunto de tareas, técnicas y métodos para la localización de un archivo.

Metadatos

Es toda aquella información descriptiva sobre el contexto, calidad, condición o características de un recurso, dato u objeto que tiene la finalidad de facilitar su recuperación, autenticación, evaluación y preservación (Méndez & Senso, 2004).

La asociación de metadatos descriptivos normalizados a los objetos de la red tiene el potencial para mejorar sustancialmente las capacidades de localización, recuperación, facilitando búsquedas basadas en campos

(p.ej autor, título), permitiendo la indización de objetos no textuales, y facilitando el acceso al contenido referenciado que es distinto del acceso al contenido del propio recurso (Perez, 2005).
En la presente investigación el autor considera que un metadato no es más que un dato que va caracterizar o describir un fichero.

Motor de búsqueda

Los buscadores de Internet o motores de búsqueda, son sistemas informáticos que dan la posibilidad de consultar una gigantesca base de datos para encontrar páginas *Web*. Los *Buscadores de Internet* brindan a los cibernautas la opción de encontrar la información que necesitan de una forma rápida, ágil y sencilla (Castillo, 2015).

Analizan más ampliamente los contenidos de las páginas *Web* y la manera en la que los usuarios acceden a ellas, analizando el tipo de consultas que se aplican o el lenguaje estudiando para obtener esta información de esta forma, sin necesidad de asociar metadatos ni dotarlos de significados semánticos, establecen las relaciones conceptuales entre las palabras utilizadas en las búsquedas y en los textos, inferirían los intereses de los usuarios sobre los contenidos de las *Web* y los procesan para ofrecer mejores resultados en las búsquedas de los usuarios (Mayer & Leisa, 2010).

En la presente investigación el autor considera que motor de búsqueda o buscador va a ser una herramienta especializada en encontrar información de una forma rápida y sencilla para ofrecerle mejores resultados de búsqueda al usuario.

1.3 Buscadores que utilizan la técnica de indexación

En el siguiente epígrafe se hace el estudio de 7 soluciones existentes que basan su búsqueda mediante diferentes métodos de Proceso de Reconocimiento Adaptativo de Patrones (APRP), redes semánticas, ecuación de ejecución simultánea e indexación de archivos. Las herramientas son Excalibur RetrievalWare, Copernic Desktop Search, Google Desktop Search, Tracker, Gnome Do, Elasticsearch y Apache Solr, que son herramientas de élite a nivel mundial en el trabajo con búsqueda e indexación de archivos.

Excalibur RetrievalWare

Líder en el mercado desde los 80 en soluciones de búsqueda y recuperación de información. Proporciona con sus herramientas funcionalidades para la gestión de la información basadas en APRP y las redes semánticas. Así, de forma simultánea es posible acceder a los patrones de información digital y al significado real de las palabras. Su objetivo es mejorar la forma en que los usuarios recuperan la información almacenada. En lugar de pedir a los usuarios que se adapten a la tecnología, esta se adapta a la forma de pensar y trabajar de las personas.

Los motores de recuperación del conocimiento *Excalibur*, lo que se denomina comúnmente "la familia *Excalibur*" son: *Excalibur RetrievalWare* para texto, *Excalibur Visual RetrievalWare* para imágenes y *Excalibur Screening Room* para formatos multimedia de las cuales se analiza *Excalibur RetrievalWare*. Los productos de *Excalibur* intentan adaptarse a tipos de datos múltiples y ofrecer soluciones para afrontar los retos provocados por la explosión informativa derivada de Internet (Albert, Riera, Abad, & Montiel, 2002).

Algunas de las características de este buscador son:

- Procura una captura más rápida y exacta del conocimiento, facilitando a los usuarios un acceso instantáneo y unas capacidades de recuperación que recorren todos los almacenes de información de una empresa, incluyendo Internet o simplemente documentación en papel.
- Integra dos nuevas formas de acceder a la información, además de la ya conocida booleana, exacta o tradicional:
  - ✓ *Patronal:* el sistema de patrones APRP, incorporado al motor, descompone cada palabra en un mapa de bits y realiza las búsquedas por medio de una comparación porcentual entre dichos mapas, evitando así cualquier posible error en el reconocimiento de caracteres.
  - ✓ *Semántica o conceptual:* gracias a su red semántica *e-lexis*, *RetrievalWare* localiza cualquier texto que contenga una palabra dada, incluyendo sus sinónimos, derivados y palabras relacionadas.
- Las aplicaciones y componentes de *RetrievalWare* hacen que el manejo de las herramientas sea fácil y de gran potencia. Los usuarios encuentran con precisión los datos requeridos mientras

navegan por bases de datos cada vez más voluminosas, bien en entornos de grupos de trabajo o en Internet.

- Proporciona capacidades de "perfilación" (difusión selectiva de información) en tiempo real.
- Búsqueda retrospectiva de imágenes documentales, de texto combinado con bases de datos o búsquedas de datos visuales a partir de su contenido.
- La arquitectura flexible y modular está diseñada para una gran adaptabilidad en entornos Cliente/Servidor y ofrece un set completo de herramientas de desarrollo de la aplicación -desde MS Visual Basic hasta las librerías C.
- Las plataformas soportadas por *RetrievalWare* incluyen sistema operativo GNU/Linux y servidores Windows/NT con clientes PC, sistema operativo GNU/Linux y clientes Internet.
- Capacidad para mejorar la productividad y reducir costes en la explotación de los sistemas de gestión, ya que elimina los pre-procesamientos de datos y los costes de aprendizaje del usuario (Albert, Riera, Abad, & Montiel, 2002).

Copernic Desktop Search

Este buscador es un producto de la empresa canadiense *Copernic Technologies*, es un agente buscador que transfiere una ecuación de búsqueda a un conjunto de buscadores de manera simultánea, recupera las referencias pertinentes y las ordena según el grado medio de relevancia obtenido de cada uno de los buscadores.

Copernic Desktop Search es un buscador de ficheros de todo tipo en el ordenador. Es una herramienta que posee un potente motor de búsqueda. Recorre con rapidez el disco duro en busca de ficheros del tipo: *Microsoft Word, Excel, y PowerPoint,* Formato de Documento Portátil (*PDF*), música en todos sus formatos, imágenes y videos también en todos sus formatos. Realiza las búsquedas tanto en discos duros como en cualquier otra unidad. También busca en los diferentes navegadores de Internet que tenga instalados tanto en el historial, favoritos y contactos. Se puede seleccionar búsqueda por títulos y por formatos.

Es un programa que se actualiza constantemente, determinando así la ubicación de todos los ficheros que estén almacenados en el disco duro anexándolos a su potente memoria, permitiendo de esta manera la búsqueda de un modo sencillo y a una gran velocidad.

Posee una interfaz que es fácil de usar, utiliza un sistema de índice que va agregando cada fichero nuevo a medida que se guarda en el disco. Este índice es personalizable, es decir, es posible recoger en él solo un tipo determinado de ficheros con la información que más interese tener disponible a la hora de realizar búsquedas (Franganillo & Figuerola, 2001).

Google Desktop Search

Es una aplicación de búsqueda que permite acceder fácilmente a la información almacenada en el equipo y en la *Web*. Con esta se puede buscar información en los mensajes de correo electrónico, archivos, música y fotografías. Indexa de forma automática prácticamente cualquier tipo de archivo y permite buscar el texto completo de estos. Entre los tipos de archivo que indexa el programa, se incluyen los siguientes: archivos de texto y código fuente, archivos *PDF*, archivos de Lenguaje de Marcas de Hipertexto (*HTML*), documentos OpenOffice.org y páginas de información, nombres de archivos y carpetas, *Microsoft Word, Excel y PowerPoint.*

Google Desktop Search empieza a indexar los archivos del equipo inmediatamente después de haberlo instalado. Esta indexación, que tiene lugar una sola vez, se ha diseñado para que coexista perfectamente con el trabajo diario, así que se puede continuar trabajando. En función de los archivos y de los otros elementos que existan en el equipo, este proceso puede durar varias horas. Cuando finaliza, *Google Desktop Search* se asegura de que el índice esté actualizado: agrega los mensajes de correo electrónico que se van recibiendo, los archivos que se actualizan y las páginas *Web* visitadas. Está disponible en varios idiomas. Se adapta automáticamente al idioma definido en la configuración del equipo (Echevarría, 2008).

Tracker

Es un sistema que automáticamente extrae información útil y metadatos de todos los ficheros, de forma que solo tiene que recordar algo de información asociada a un fichero para poder encontrarlo: el nombre de un

álbum, una palabra de un texto, el asunto de un correo electrónico. Es capaz de buscar y encontrar información, no solo nombres de ficheros.

Su funcionamiento se basa en tener una capa entre el usuario y todos los ficheros almacenados en el disco duro de la máquina, de forma que todas las aplicaciones que estén integradas con Tracker puedan acceder a esa información detallada y ordenada e intentar dejar un poco de lado la estructura convencional de directorios y búsqueda de ficheros (Latorre, 2008).

Algunas de las características de este buscador son:

- Indexar también el contenido de los ficheros.
- Generación automática de miniaturas de los archivos.
- Indexar los dispositivos montados.
- Especificar los directorios que se quieren indexar.
- Indexar los correos electrónicos almacenados en Evolution.
- Opciones de rendimiento y consumo de recursos.

Elasticsearch

Es una potente herramienta que permite indexar gran volumen de datos y posteriormente hacer consultas sobre ellos, soportando entre otras muchas cosas: búsquedas aproximadas, facetas y resaltadas.
Elasticsearch se basa en la tecnología Lucene pero expone su funcionalidad a través de una interfaz de Transferencia de Estado Representacional (*REST*) recibiendo y enviando datos en formato de Notación de Objeto JavaScript (*JSON*) y oculta mediante esta interfaz los detalles internos de Lucene. Esta interfaz permite que pueda ser utilizada por cualquier plataforma no solo Java, puede usarse desde Python, el framework desarrollado por Microsoft (.NET), una Página Personal (PHP) o incluso desde un navegador con Javascript. Los que indexemos en ella sobrevivirá a un reinicio del servidor (Picodotdev, 2014).

Algunas de las características de este buscador son:

- Posibilita hacer búsquedas a texto completo.
- Filtrado por resaltado (highlight) y facetas (facets).
- Soporta diferentes analizadores según el idioma de la propiedad en que se busque.

- La indexación o no de los datos es responsabilidad de usuario.

Gnome Do

Es una herramienta de indexación, algunas de las características de este buscador son:

- Permite principalmente mejorar el reconocimiento de las aplicaciones del menú mostrando su nombre y descripción.
- Se ejecuta automáticamente al inicializar la computadora.
- Tiene la ventaja de que va a tener un consumo mínimo de los recursos en la Unidad Central de Procesamiento (CPU).
- Tiene la limitante de que no realiza búsquedas en contenido del fichero y para la indexación de archivos es necesario definir en un fichero de configuración las carpetas del sistema a indexar, así como el nivel al que se realizará esta indexación (Pablo, 2009).

Apache Solr

Apache Solr es una plataforma de búsquedas basada en la tecnología Apache Lucene, que funciona como un "servidor de búsquedas". Sus principales características incluyen búsquedas de texto completo, de los resultados, clustering dinámico, y manejo de documentos ricos (como *Word* y *PDF*). Solr es escalable, permitiendo realizar búsquedas distribuidas y replicación de índices, y actualmente se usa en muchos de los sitios más grandes de Internet (Kuć, 2013).

Algunas de las características de este buscador son:

- Posibilita filtrar los resultados de búsqueda mediante diversos parámetros, como cualquier metadato asociado.
- Realiza indexación parcial o completa de ficheros.
- Búsqueda en contenido.
- Configurar el esquema de datos a indexar.
- Cantidad de resultados a obtener por cada consulta realizada.
- Configurar indexación automática de URL, base de datos, archivos de texto.

- Configurar consumo de recursos del CUP.
- Permite realización de indexaciones simultáneas.
- Posibilita la integración con varios lenguajes de desarrollo.

A continuación en la Tabla 1 se muestra en la comparativa de las herramientas existentes con el objetivo de seleccionar una de estas para su posterior utilización en el proceso de implementación. Los elementos que se van a tener en cuenta son: su capacidad búsqueda no solo en los metadatos básicos sino también en lo que contiene el fichero en su interior. La búsqueda puede ser realizada mediante el filtrado de datos, lo que posibilitará a los usuarios buscar por un criterio específico, ya sea nombre, autor o contenido. Es necesario tener en cuenta el consumo del CPU en cuanto a las funcionalidades con las que debe cumplir la solución; ya que un consumo mínimo estará dado por debajo de 512MB de memoria RAM (memoria de acceso aleatorio), un consumo medio entre 512MB y 1GB y un consumo alto más de 1GB. Se debe verificar la presencia de filtros para los resultados devueltos en una búsqueda para una vez obtenida la misma, reducir la cantidad de resultados. Las aplicaciones deben realizar búsquedas locales para comprobar si se puede utilizar localización de ficheros locales.

	Excalibur RetrievalWare	Copernic Desktop Search	Google Desktop Search	Tracker	Elasticsearch	Gnome Do	Apache Solr
Búsqueda en contenido	x		x	X	X		X
Filtrado de datos	x	x	x	X	X	X	X
Consumo mínimo de					X	X	X

recursos del CPU							
Consumo medio de recursos del CPU	x	x	x		X		X
Consumo máximo de recursos del CPU				X	X		X
Filtros de búsqueda	x	X	x	X	X	X	X
Búsqueda local	x	X	x	X	X	x	X

Tabla 1: Comparación de las herramientas de búsqueda.

1.3.1 Análisis de los sistemas homólogos

Los sistemas estudiados presentan funcionalidades y características que aportan beneficios a la solución, pero por separado no satisfacen todas las expectativas o presentan limitantes para el uso pleno de sus funcionalidades. Se espera que el sistema sea capaz de indexar cualquier tipo de archivo y realizar búsqueda tanto en los metadatos como en el contenido y posibilite la gestión de los mismos. En el caso de *Excalibur RetrievalWare,* es una herramienta que solo se va a limitar a la búsqueda en documentos y no así en otros ficheros del sistema. *Copernic Desktop Search* es un buscador que se centra en los buscadores del sistema, ejecutando varias búsquedas simultáneas pero no es capaz de analizar el contenido, por lo que se pueden ignorar resultados de interés para el usuario. *Google Desktop Search* a pesar de cumplir con todas las expectativas tiene la limitante de que para su uso pleno dependerá de varias Interfaces de

Programación de Aplicaciones (API´s) del motor de *Google,* que a causa del bloqueo injusto que lleva los Estados Unidos contra Cuba es imposible acceder a ellas. Tracker es una potente herramienta de indexación que posibilita la búsqueda en contenido, se centra en romper con la estructura básica de directorios organizando y catalogando la información por tipo; a pesar de esto y de poder ser utilizada la misma va a tener un alto consumo de recursos del CPU tanto de la RAM como del Microprocesador. *Elasticsearch* va ser otra de las herramientas que realizan su búsqueda basada en la indexación, la cual proporciona un gran conjunto de funcionalidades para la indexación de grandes volúmenes de datos, pero no se decide utilizarla debido a que se tomará para utilizar en el desarrollo su homóloga Apache Solr que presenta un modelo más amplio en cuanto a tipos de ficheros a analizar y formatos. Es una potente herramienta de búsqueda e indexación, la cual presenta un gran conjunto de funcionalidades que facilitan la realización de búsquedas e indexación así como posibilita indexar grandes volúmenes de datos y realiza un análisis total de los metadatos asociados a un fichero. Por lo que se llega a la conclusión de utilizar las funcionalidades de Apache Solr como motor de búsqueda e indexación en el desarrollo de una herramienta nacional, capaz de realizar búsquedas mediante indexación y en contenido para la Distribución Cubana de GNU/Linux Nova Escritorio 5.0.

1.4 Tecnologías utilizadas para la construcción de la solución

A continuación se describen la metodología y las principales herramientas, tecnologías y lenguajes, seleccionados dado un análisis de su integración y compatibilidad con el motor de búsqueda.

1.4.1 Metodología de desarrollo de software

Una metodología de desarrollo es un proceso de software detallado y completo. Se basa en una combinación de los modelos de procesos genéricos. Define artefactos, roles y actividades involucradas, junto con prácticas y técnicas recomendadas (Pressman, 2005).

Como metodología de desarrollo se decide usar **OpenUp**, ya que es la metodología definida por el departamento Sistema Operativo (SO) para el desarrollo de soluciones informáticas de apoyo a la Distribución Cubana de GNU/Linux Nova.

Esta metodología adopta un enfoque pragmático, con una filosofía ágil que se centra en la naturaleza colaborativa del desarrollo de software. Es una herramienta agnóstica, un proceso de baja formalidad que puede ser usado tal cual o ampliarse para hacer frente a una amplia variedad de proyectos.

La misma va a estar basada en los siguientes principios:

- ✓ Colaborar para sincronizar intereses y compartir conocimiento. Este principio promueve prácticas que impulsan un ambiente de equipo saludable, facilitan la colaboración y desarrollan un conocimiento compartido del proyecto.
- ✓ Equilibrar las prioridades para maximizar el beneficio obtenido por los interesados en el proyecto. Este principio promueve prácticas que permiten a los participantes de los proyectos desarrollar una solución que maximice los beneficios obtenidos por los participantes y que cumpla con los requisitos y restricciones del proyecto.
- ✓ Centrarse en la arquitectura de forma temprana para minimizar el riesgo y organizar el desarrollo.
- ✓ Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo. Este principio promueve prácticas que permiten a los equipos de desarrollo obtener retroalimentación temprana y continua de los participantes del proyecto, permitiendo demostrarles incrementos progresivos en la funcionalidad (Juarez, 2013).

1.4.2 Lenguajes utilizados

Un lenguaje de programación se usa para traducir las clases, atributos, operaciones y mensajes, de manera que puedan ejecutarse por la máquina (Pressman, 2005).

Se define **Java** como lenguaje de programación a utilizar, dado que para el desarrollo del sistema el motor de búsqueda seleccionado define un conjunto de librerías para desarrollo de soluciones de escritorio que se encuentran implementadas en este lenguaje; lo que va a permitir una correcta integración. A continuación se enuncian características del lenguaje seleccionado.

Es un lenguaje de programación desarrollado por *Sun Microsystems* a principios de los años 90. Trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. La tecnología Java está compuesta básicamente por 2 elementos: el lenguaje y su plataforma, la máquina virtual de *Java (Java Virtual Machine)*.

Este lenguaje fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Los programadores experimentados en *C++* pueden migrar muy rápidamente a *Java* y ser productivos en poco tiempo, posee una curva de aprendizaje muy rápida (Gil, 2004).

El motor de búsqueda va a constar con una interfaz visual administrativa vía *Web*, por lo que se define el lenguaje ***HTML*** para su uso en el servidor. A continuación se muestran características de este lenguaje.

Es un lenguaje de composición de documentos y especificación de etiquetas que define la sintaxis y coloca instrucciones especiales que no muestra el navegador, aunque si le indica cómo desplegar el contenido del documento, incluyendo texto, imágenes y otros medios soportados. Indica cómo hacer un documento interactivo a través de etiquetas especiales, las cuales conectan diferentes documentos y otros recursos de Internet.

Este también permite ciertos códigos que se conocen como scripts, los cuales brindan instrucciones específicas a los navegadores que se encargan de procesar el lenguaje. Entre los scripts que pueden agregarse, los más conocidos y utilizados son JavaScript y PHP.

El marcado estructural es el que estipula la finalidad del texto, aunque no define cómo se verá el elemento. El marcado presentacional, por su parte, es el que se encarga de señalar cómo se verá el texto más allá de su función (Morón, Yajaira, Yépez, & Yaimileth, 2004).

Para el desarrollo se define también **Metalenguaje de Etiquetado Extensible** (XML), para la configuración vía archivo del motor de búsqueda y los formatos de ficheros a indexar por el sistema son definidos por

motor de búsqueda en este formato; también posibilita la definición del modelo de datos al realizar la indexación de los ficheros locales, y es el formato en que se devuelve la información indexada. A continuación se enuncian sus características:

El XML fue creado por el *World Wide Web Consortium* (W3C) a finales de los 90. El W3C se creó en 1994 para tutelar el crecimiento y organización de la *Web*. Su primer trabajo fue normalizar el *HTML*, el metalenguaje de marcas con el que se escriben las páginas *Web*. Al crecer el uso de la *Web*, crecieron las presiones para ampliar el *HTML*. El W3C decide que la solución no era ampliar el *HTML*, sino crear unas reglas para que cualquiera pudiera crear lenguajes de marcas adecuados a sus necesidades, pero manteniendo una estructura y sintaxis comunes que permitieran compatibilizarlos y tratarlos con las mismas herramientas. Ese conjunto de reglas es el XML, cuya primera versión se publicó en 1998 (Marco, 2014).

XML se ha creado con el fin de enriquecer la estructura de los documentos que pueden ser usados en la *Web* los objetivos de diseñar XML fueron los siguientes:

- XML debe ser directamente utilizable en Internet.
- XML debe soportar una amplia variedad de aplicaciones.
- XML debe ser compatible con Lenguaje de Marcado Generalizado Estándar (SGML).
- Debería ser sencillo escribir programas que procesaran documentos XML.
- El número de las características opcionales en XML debería ser el mínimo posible, a ser posible cero.
- Los documentos XML deberían ser legibles por las personas y razonablemente claros.
- El diseño de XML debe ser rápido.
- XML debería ser simple, pero perfectamente normalizado.
- Los documentos XML deben ser de fácil creación.
- La concisión de las marcas XML tiene una importancia mínima.

Como parte del proceso de desarrollo se define el **Lenguaje Unificado de Modelo** (UML) lenguaje utilizado por la herramienta Visual Paradigm utilizada para el diseño de los diagramas correspondientes al ciclo de vida de desarrollo de la metodología. El mismo presenta las siguientes características.

El UML se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software. Es un sistema notacional (que, entre otras cosas, incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos (Larman, 1999).

Características UML:

- Va a ser tener un lenguaje con sintaxis y artefactos de relaciones y diagramas.
- Es usa para el modelado de: diagrama de clases, diagrama de objetos, casos de uso, componentes, secuencia, colaboración, estados y actividades.

1.4.3 Herramienta de modelado

Para el modelado de los diagramas generados a lo largo del ciclo de vida del proyecto, se hace necesario utilizar una herramienta de Ingeniería de Software Asistida por Computadora (*CASE)*. Las herramientas *CASE* son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida del desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores.

Entre sus principales características se encuentran:

- Entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.

- Disponibilidad de múltiples versiones, para cada necesidad.
- Generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos.

Se define **Visual Paradigm 8** para la generación de los diagramas. Es la herramienta *CASE* utilizada en la universidad para el modelado debido a que en su versión libre no se necesita estar pendiente de que caduque o se pague por su licencia.

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite modelar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación (Larman, 1999).

1.4.4 Motor de búsqueda de indexación

Un motor de búsqueda, también conocido como buscador, consiste básicamente en un sistema informático que es capaz de realizar la búsqueda a gran velocidad de un archivo que esté alojado en cualquier servidor. **Apache Solr** en su versión 4.10.2 va a ser el motor de búsqueda e indexación a utilizar el mismo va presentar una perfecta integración con varios framework de desarrollo *Web* como son Symfony, *Word*Press y Drupal, así como presenta un conjunto de librerías diseñadas para su uso en el desarrollo herramientas de escritorio con soporte JAVA. A continuación se presentan las características del sistema.

Apache Solr está escrito en Java y se ejecuta como un servidor de búsqueda completo independiente dentro de un contenedor *servlets* como *Jetty*. *Solr* utiliza la librería de búsqueda Java Lucene en su núcleo para indexar y buscar texto completo, y tiene un HTTP/XML similar a *REST* y APIs *JSON,* lo cual facilita su uso desde virtualmente cualquier lenguaje de programación. La poderosa configuración externa de Solr permite adaptarse a casi cualquier tipo de aplicación sin codificación Java, y tiene una amplia arquitectura de *plugins* y cuando se requieren, personalizaciones más avanzadas (Apache Solr, 2014).

1.4.5 Protocolo de Internet

Un protocolo es un método estándar que permite la comunicación entre procesos (que potencialmente se ejecutan en diferentes equipos), es decir, es un conjunto de reglas y procedimientos que deben respetarse para el envío y la recepción de datos a través de una red. Existen diversos protocolos de acuerdo a como se espera que sea la comunicación (Kioskea, 2015).

El sistema de indexación de archivos de la Distribución Cubana de GNU/Linux Nova Escritorio 5.0 es una aplicación de escritorio, pero emplea **Protocolo de Transferencia de Hipertexto** (*HTTP)* para realizar las peticiones al servidor de indexación.

Es un sencillo protocolo cliente-servidor que articula los intercambios de información entre los clientes *Web* y los servidores *HTTP*. Fue propuesto por Tim Berners-Lee, atendiendo a las necesidades de un sistema global de distribución de información como el *WWW* (*World Wide Web*).

Se basa en sencillas operaciones de solicitud/respuesta. Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud. El servidor responde con un mensaje similar, que contiene el estado de la operación y su posible resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan; cada objeto *Web* es conocido por su URL (Laguillo, 1998).

Características:

- Todas las comunicaciones entre los clientes y los servidores se realizan a partir de caracteres US-ASCII de 7 bits.
- Permite la transferencia de objetos multimedia.
- Cada operación HTTP implica una conexión con el servidor.
- No mantiene estado.

1.4.6 Plataforma de desarrollo

Una plataforma de desarrollo es el entorno de software común en el cual se desenvuelve la programación de un grupo definido de aplicaciones. Comúnmente se encuentra relacionada directamente a un sistema

operativo; sin embargo, también es posible encontrarla ligada a una familia de lenguajes de programación o a una API, también funciona como sistema plataforma o multiusuario (Suela, 2013).

Para la ejecución del sistema se define el **Kit de Desarrollo de Java** (JDK) en su versión 1.7, el mismo contiene las herramientas y librerías necesarias para crear y ejecutar applets y aplicaciones en Java (CEC, 2013).

A continuación se listan algunas de las utilidades que se pueden encontrar en el JDK:

- ***Javac***. Es el compilador de Java. Se encarga de convertir el código fuente escrito en Java a *bytecode*.
- ***java***. Es el intérprete de Java. Ejecuta el *bytecode* a partir de los archivos class.
- ***appletviewer***. Es un visor de applets. En la mayoría de las ocasiones puede utilizarse en lugar de un Navegador *Web*.
- ***javadoc***. Se utiliza para crear documentación en formato *HTML* a partir del código fuente Java y los comentarios que contiene.
- ***javap***. Es un desensamblador[1] de Java.
- ***jar***. Es una herramienta utilizada para trabajar con los archivos *JAR*.

1.4.7 Entorno de Desarrollo Integrado

Un Entorno de Desarrollo Integrado (IDE) es un programa compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien, poder utilizarse para varios. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes.

[1] es un programa de computador que traduce el lenguaje de máquina a lenguaje ensamblador.

Se define como IDE ***NetBeans*** en su versión 8. Es uno de los más usado para el desarrollo de aplicaciones escritas en el lenguaje JAVA.

NetBeans es un entorno de desarrollo multilenguaje, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el *NetBeans* IDE. Es un producto libre y gratuito sin restricciones de uso. Tiene soporte para crear interfaces gráficas de forma visual, crear aplicaciones para móviles, desarrollar aplicaciones *Web* y además estas funcionalidades son ampliables mediante la instalación de paquetes adicionales (NetBeans, 2015).

NetBeans IDE se conoce como la solución más completa para programar en Java. Se encuentra disponible para *Windows*, *Mac*, *Linux* y *Solaris*.

Características de *NetBeans*:

- ✓ Buen editor de código.
- ✓ Soporte para *Ruby*, *JRuby*, y *Ruby on Rails*.
- ✓ Instalación y actualización más simple.
- ✓ Enlazar datos con el *Swing GUI*.
- ✓ Características visuales para el desarrollo *Web*.
- ✓ Creador gráfico de juegos para celulares.
- ✓ Mejoras para *SOA (Arquitectura Orientada a Servicios)* y *UML*.
- ✓ Soporte para *PHP*.

1.4.8 Servidor *Web*

Un servidor *Web* es un programa que implementa el protocolo *HTTP*. Este protocolo está diseñado para transferir hipertextos, páginas *Web* o páginas *HTML*: textos complejos con enlaces, figuras, formularios,

botones y objetos incrustados como animaciones o reproductores de música. Este servidor se encarga de mantenerse en espera de las peticiones *HTTP* llevadas a cabo por un cliente. Este realiza una petición al servidor, quien le responde con el contenido solicitado.

Es el servidor utilizado para el despliegue e instalación del Apache Solr. ***Tomcat*** es un contenedor *Web* con soporte de *servlets* y Servidor de Páginas de Java (*JSPs*). *Tomcat* no es un servidor de aplicaciones, como *JBoss* o *JOnAS*. Incluye el compilador *Jasper*, que compila *JSPs* convirtiéndolas en *servlets*.

En sus inicios existió la percepción de que el uso de *Tomcat* de forma autónoma era solo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y *Tomcat* es usado como servidor *Web* autónomo en entornos con alto nivel de tráfico y alta disponibilidad.

Dado que *Tomcat* fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java (Alvarez, 2009).

1.5 Conclusiones parciales

En el presente capítulo se abordan los principales conceptos y definiciones que contribuyen a la comprensión de la investigación. Así como también el análisis de los homólogos tales como *Excalibur RetrievalWare, Copernic Desktop Search y Google Desktop Search, Tracker, Elasticsearch, Gnome Do y Apache Solr* que emplean entre sus técnicas la indexación de archivos propuesta, concluyendo que de las mismas solo se utilizará para el desarrollo de la solución Apache Solr que presenta todo el conjunto de funcionalidades y requisitos que ofrecen una solución a la problemática de la investigación.

Se define la metodología *OpenUP* para guiar el proceso de desarrollo de software, así como el lenguaje de programación *Java*, el lenguaje de marcas de hipertexto *HTML* y el lenguaje de marcas extensible *XML*. Se selecciona *Apache Solr 4.10.2* como motor de búsquedas y *Tomcat 7* como servidor *Web*. Como entorno de desarrollo integrado se decide utilizar *NetBeans* 8 y para el modelado de los diagramas propuestos se empleará el *Visual Paradigm* 8.0.

Capítulo 2: Propuesta de Solución

2.1 Introducción

El proceso de análisis para el desarrollo de software es una tarea fundamental para la futura implementación, por lo que es necesario definir bien las acciones a realizar y los resultados que se esperan para determinar las características a cumplir por el sistema.

2.2 Propuesta de solución

Para dar solución al problema existente de la búsqueda de información en la Distribución Cubana de GNU/Linux Nova Escritorio 5.0, se propone el desarrollo de una aplicación que permitirá a los usuarios realizar búsquedas en el sistema, basándose no solo en sus metadatos sino también en el contenido de los mismos. Entre las características principales del sistema además, se persigue que sea capaz de realizar búsquedas simples o basadas en índices, filtrar el domino de búsqueda, configurar la conexión al motor de búsqueda basado en índices y gestionar archivos.

El sistema constará de un entorno de escritorio para la realización de las búsquedas, configuración de las principales características del motor de búsqueda a utilizar, así como también una interfaz *Web* para la configuración avanzada de las propiedades de Apache Solr. Para un mayor entendimiento de la propuesta de solución se presenta un diagrama de dominio donde se exponen las clases conceptuales de la propuesta.

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les denomina modelos conceptuales (término utilizado en la primera edición del libro de Laman), modelo de objetos del dominio y modelos de objetos de análisis (Hall, 2003).

Descripción de las clases conceptuales del dominio

NovaBuscador: objeto encargado de contener las clases a utilizar por el buscador.

Controladora: objeto encargado de crear las instancias pertenecientes a las clases para realizar las búsquedas.

Configuración de Conexión: objeto encargado de crear, cambiar y cerrar una conexión con el motor de búsqueda.

Librería JSolr: conjunto de objetos encargados de propiciar todas las clases y funciones necesarias para la comunicación con el servidor de indexación.

Apache Solr: motor de búsqueda a utilizar para la búsqueda mediante índices y contenidos en los ficheros.

Resultados de la búsqueda: son los valores obtenidos en la realización de una petición.

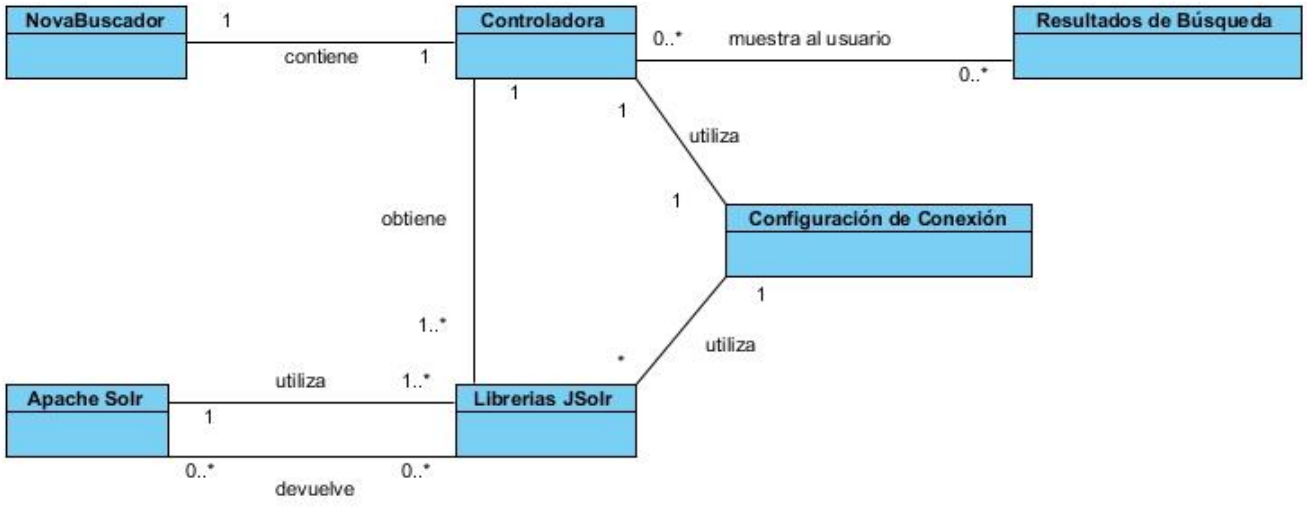


Figura 1: Modelo de dominio.

En la Figura 1 se muestran las relaciones del modelo asociadas a la propuesta de solución planteada. El objeto NovaBuscador va a ser el objeto encargado de contar con las clases necesarias para el funcionamiento del sistema, el mismo presenta contener una controladora que va a ser la encargada del control total de los métodos para realizar la búsqueda; obtiene funcionalidades de las librerías JSolr y Configuración de Conexión que proporcionarán los mecanismos necesarios para la comunicación con el

motor de búsqueda e indexación Apache Solr y finalmente poder obtener uno o varios resultados de búsqueda.

2.3 Características del sistema y cualidades

Requisitos Funcionales (RF)

Los RF son capacidades o condiciones que el sistema debe cumplir. Se mantienen invariables sin importar con qué propiedades o cualidades se relacionen y muestran como los casos de uso serán llevados a la práctica (Pressman, 2005).

Para el desarrollo de la aplicación se identifica un total de 11 RF, los que se muestran a continuación en la Tabla 2.

#	RF
1	Configurar conexión con el servidor
2	Acceder a carpeta local
3	Abrir fichero
4	Indexar ficheros
5	Realizar búsqueda
6	Realizar búsqueda mediante el uso de índices
7	Realizar búsqueda en contenido de ficheros
8	Mostrar el sistema de archivos de la PC
9	Mostrar el valor de los metadatos de un fichero
10	Mostrar resultados de la búsqueda
11	Filtrar el domino de búsqueda según un criterio

Tabla 2: Requisitos funcionales del sistema.

Requisitos No Funcionales (RNF) del sistema

Los RNF describen aspectos del sistema que no incluyen una relación directa con el comportamiento funcional del sistema, estos incluyen restricciones como el tiempo de respuesta, precisión recursos consumido y seguridad (Quiroga, 2014).

A continuación se enuncian los RNF del sistema a desarrollar:

Hardware

RNF1. Un microprocesador Celeron, con una velocidad mínima de 1.3 GHz, 512 MB de capacidad de disco duro para su despliegue, 512 MB RAM.

Software

RNF2. Apache Tomcat 1.7, *Apache Solr* 4.10.2 o superior, *Java JDK* 1.7 o superior, *Java JRE* 1.7 o superior.

Funcionalidades del sistema

Un Caso de Uso (CU) es una secuencia de interacciones entre un sistema y alguien o algo que usa alguno de sus servicios (Ceria, 2001).

A continuación se presenta el diagrama de CU de la aplicación, el cual constará con un total de 6.

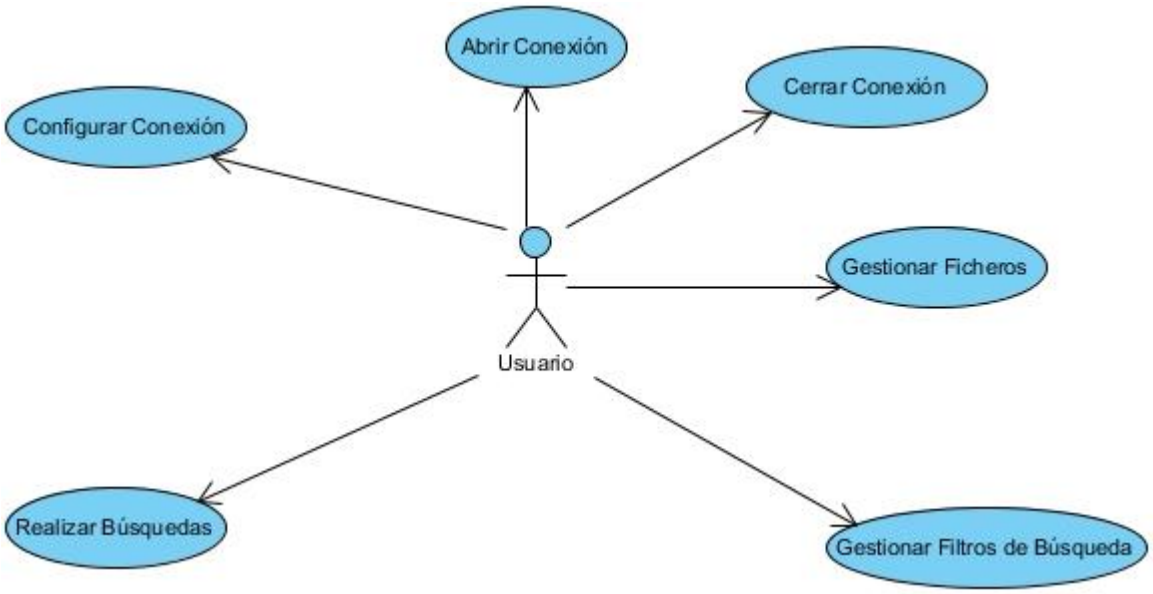


Figura 2: Diagrama de CU del sistema.

El sistema será utilizado por un único usuario que abre y cierra la conexión con el servidor de indexación, el CU 1 Configurar Conexión posibilita cambiar la dirección de conexión con el servidor en caso de un cambio en la configuración por defecto. El CU 2 Gestionar Ficheros posibilita ver los ficheros existentes en la PC así como acceder a los mismos. El CU 3 Realizar Búsqueda es el encargado de enviar las solicitudes al servidor e indexar los ficheros. Por último el CU 4 Gestionar Filtros de Búsquedas permite seleccionar los filtros a la hora de realizar una búsqueda específica.

Especificación de CU del sistema

La especificación de los CU se refiere a la descripción de cada uno de los elementos que detallan los CU, su descripción completa se logra bajo un formato de tabla con los campos: **Objetivos**, **Actores**, **Resumen**, **Complejidad**, **Prioridad**, **Precondiciones**, **Postcondiciones**, **Flujo de eventos** (describe las acciones del **Usuario** y del **Sistema** en pasos que se sucederían en el escenario “básico”; es un camino simple, sin ramificaciones y en él suelen hacerse una serie de asunciones) **y Flujos alternos**, que constituyen las alternativas del flujo de eventos básico.

Seguidamente se presenta la descripción correspondiente al CU “Configurar conexión”, ya que el mismo constituye un punto clave en la implementación de la misma; es imprescindible para el correcto funcionamiento de la búsqueda basada en indexación, ya que en ausencia de una conexión con el servidor de búsqueda, el sistema solamente podrá realizar búsquedas básicas. Es el punto de partida para poder realizar cualquier búsqueda basada en el índice. (Ver el resto de las descripciones en anexos).

CU Configurar Conexión

Objetivo	Configurar Conexión	
Actores	Usuario	
Resumen	Posibilita cambiar la dirección de conexión al servidor de indexación	
Complejidad	Muy Alta	
Prioridad	Crítico	
Precondiciones		
Postcondiciones		
Flujo de eventos		
	Actor	**Sistema**
1.	Hace clic en el menú de opciones en la opción Configurar Conexión.	Muestra una vista con el campo: - Nueva Dirección.

2.	Introduce la dirección. Pulsa la opción “Guardar”.	Verifica la conexión. (Si ocurre un error, sucede el flujo alterno 1)
3.		Crea una conexión.
4.		Muestra un mensaje de conexión satisfactoria.
5.		Termina el Caso de Uso.
Flujos alternos		
1 Error en la conexión.		
	Actor	Sistema
1.		No se crea la conexión.
2.		Muestra mensaje de error al conectar.

2.4 Arquitectura de la solución

Dividir un software en varias partes lógicas, ya sean módulos, paquetes o capas, ofrece la posibilidad de comprender fácilmente su filosofía y distribuir las tareas que ejecuta. Por ello la comunidad del software desarrolló la noción de una arquitectura de varios niveles y entre las más difundidas se encuentra la arquitectura N capas.

La separación entre la lógica de la aplicación y la interfaz de usuario ofrece mayor flexibilidad al diseño de la misma. De manera que los modelos N capas están encaminados a maximizar aspectos importantes dentro de las aplicaciones, su autonomía, confiabilidad, disponibilidad, escalabilidad e interoperabilidad (Hernández & Díaz, 2009).

Para el desarrollo de la propuesta de solución se utilizará una arquitectura N Capas en su estilo arquitectónico 3 capas. Se divide la aplicación en tres partes lógicas, con un grupo de interfaces perfectamente definidas.

Presentación: Es la correspondiente a la interfaz visual de la solución que permite al usuario interactuar con las funcionalidades de realizar búsquedas, gestionar filtros de búsquedas, configurar la conexión así como las de gestionar ficheros.

Modelo: Contiene lo correspondiente a los ficheros que definen el modelo de datos a utilizar para realizar la indexación, así como las clases necesarias para interactuar mediante el servicio que ofrece las librerías de jsolr con el motor de búsqueda e indexación.

Controlador: Por último esta será la capa controladora que contiene las clase Controladora y ConfigConexion que se encargan de contener y controlar los métodos necesarios para el correcto funcionamiento de la solución.

Diagrama de paquetes

Un modelo del dominio puede crecer fácilmente y llegar a ser lo suficientemente amplio para que sea conveniente dividirlo en paquetes que incluyen conceptos fuertemente relacionados, esto sirve de ayuda para mejorar la comprensión y para abordar el trabajo de análisis en paralelo, en el que diferentes personas realizan el análisis del dominio en diferentes subdominios.

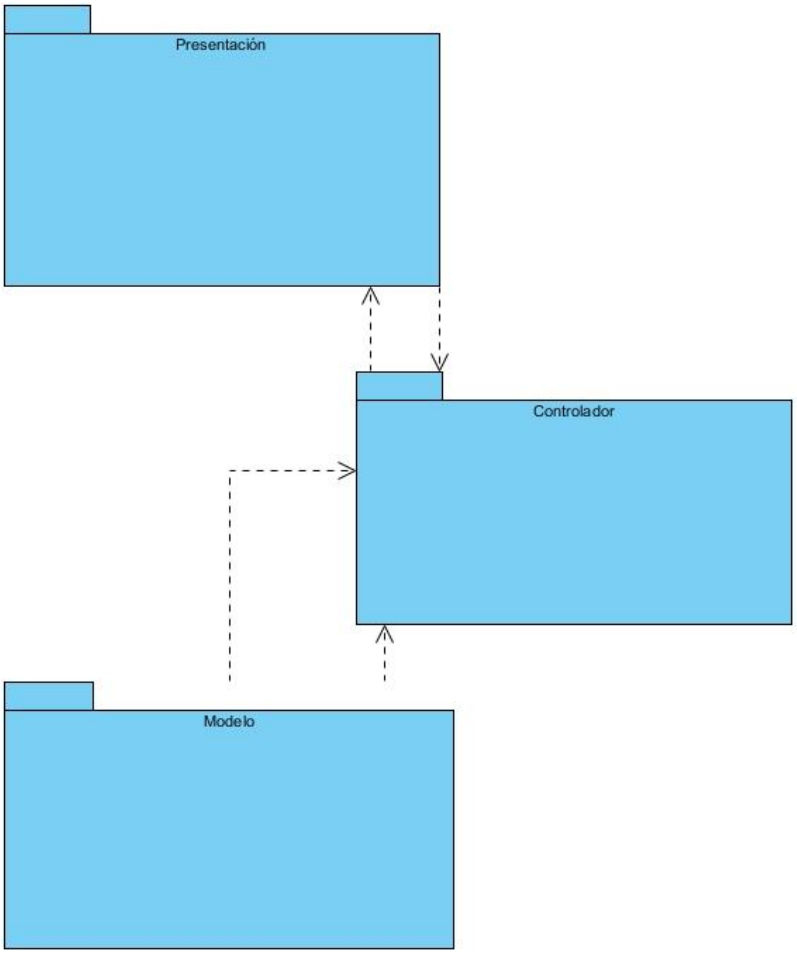


Figura 3: Diagrama de paquetes.

En el primer paquete (Presentación) paquete se encuentran las interfaces correspondientes a la aplicación; en el segundo paquete (Modelo) se alojan las librerías necesarias para realizar la búsqueda y la conexión con el servidor, así como los modelos de datos y configuraciones del motor de búsqueda, en el tercer paquete (Controlador) se hallan alojadas las clases que permitirán el uso de las funcionalidades del sistema.

2.5 Diseño de la solución propuesta

Diagrama de clases

Los diagramas de clases del diseño permiten describir gráficamente las especificaciones de las clases del software. Muestran las clases (descripciones de objetos que comparten características comunes) que componen el sistema y como se relacionan entre si (Pressman, 2005).

La Figura 4 muestra el diagrama de clases del diseño correspondiente a la propuesta de solución.

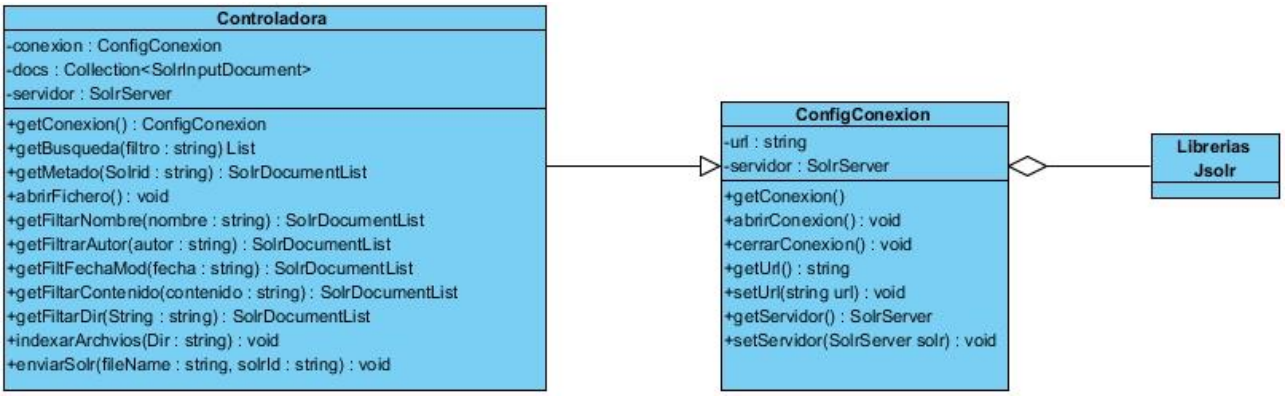


Figura 4: Diagrama de Clases del sistema.

En el anterior diagrama de clases del diseño se definen las clases involucradas. El sistema va a contar con dos clases y una librería, primeramente se encuentra la clase *Controladora*, la cual se encarga de contar con todas las funcionalidades del sistema, así como realizar instancias de otras. *ConfigConexion* se encarga de configurar y crear la conexión con el motor de búsqueda, haciendo uso de las librerías *Jsolr*, que no es más que un conjunto de clases diseñadas para establecer la comunicación, administración y configuración de motor de búsqueda Apache Solr.

2.5.1 Patrones de diseño utilizados en la propuesta de solución

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Se deben tener presentes los siguientes elementos de un patrón: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios) (Kabytes, 2010).

En la solución se destacan los Patrones Generales de Software para Asignación de Responsabilidades (GRASP) como el experto, creador y controlador. Así como también los patrones *Grang of Four* (Gof), entre los que figuran tres clasificaciones diferentes: de comportamiento, estructurales y de diseño. En la solución se evidencia el uso del patrón de diseño singleton.

Patrones GRASP

El patrón Experto va a asignar una responsabilidad al experto en información: la clase que posee la información necesaria para cumplir con la responsabilidad (Larman, 2003).

El mismo se va encontrar presente en la clase controladora, que va ser la experta en tener toda la información para el sistema.

```
import org.apache.solr.client.solrj.request.AbstractUpda
import org.apache.solr.client.solrj.request.ContentStrea
import org.apache.solr.client.solrj.response.QueryRespon
import org.apache.solr.common.SolrDocumentList;
import org.apache.solr.common.SolrInputDocument;

/**
 *
 * @author yoandri
 */
public class Controladora {

    private Collection<SolrInputDocument> docs;
    SolrServer servidor;
    ConfigConexion conexion;
    public Controladora() {

        this.docs = new ArrayList<SolrInputDocument>();
        servidor = getConexion().getServidor();
    }

    public ConfigConexion getConexion() {
         this.conexion = new ConfigConexion();

        return this.conexion;
    }
```

Figura 5: Patrón Experto en el sistema.

El patrón **Creador** se encarga de asignar a la clase B la responsabilidad de crear una instancia en la clase A, si se cumple una de las siguientes condiciones (Larman, 2003).

1. B contiene A.
2. B agrega A.
3. B tiene los datos de inicialización de A.
4. B registra A.

5. B utiliza A muy de cerca.

Este patrón se va encontrar presente en la clase ConfigConexion, ya que la misma va a ser la responsable de inicializar las clases proporcionadas por la librería *Jsolr.*

```
public ConfigConexion() {
        Url="http://localhost:8080/solr/";
        servidor = new HttpSolrServer(Url);

}
```

Figura 6: Patrón Creador en el sistema.

El patrón Controlador asigna la responsabilidad de crear una clase de eventos del sistema a una clase que representa una de las siguientes opciones (Larman, 2003):

1. El negocio o la organización global (un controlador de fachada).
2. El sistema global (un controlador de fachada).
3. Un ser animado del dominio que realiza el trabajo (un controlador de papeles).
4. Una clase artificial (Fábrica Pura) que representa el caso de uso (un controlador de caso de uso).

```
public SolrDocumentList getBusqueda(String nombre ) throws SolrServerException{
SolrQuery query = new SolrQuery();
query.setQuery( "id: "+"*"+nombre+"*");
query.addSortField( "price", SolrQuery.ORDER.asc );
QueryResponse rsp = getConexion().getServidor().query( query );
SolrDocumentList docs = rsp.getResults();

return docs;
}
```

Figura 7: Patrón Controlador en el sistema.

Patrones Gof

El patrón Singleton permite exactamente una instancia de una clase. Los objetos necesitan un solo punto de acceso (Larman, 2003).

El mismo se encuentra presente en la creación de la conexión con el servidor de indexación, ya que solamente se podrá crear una única instancia de esta una vez que se encuentre abierta.

```
public ConfigConexion getConexion() {
    this.conexion = new ConfigConexion();

    return this.conexion;
}
```

Figura 8: Patrón Singleton en el sistema.

Diagrama de Secuencia del CU Realizar búsqueda

El diagrama de secuencia muestra los objetos de un escenario mediante líneas verticales y los mensajes entre los objetos conectados con flechas; mientras que los mensajes son dibujados cronológicamente de arriba hacia abajo. A continuación, en la figura se muestra el diagrama de secuencia correspondiente al CU "Realizar búsqueda". En el mismo se ilustra el flujo secuencial por el que pasa el sistema para realizar una búsqueda.

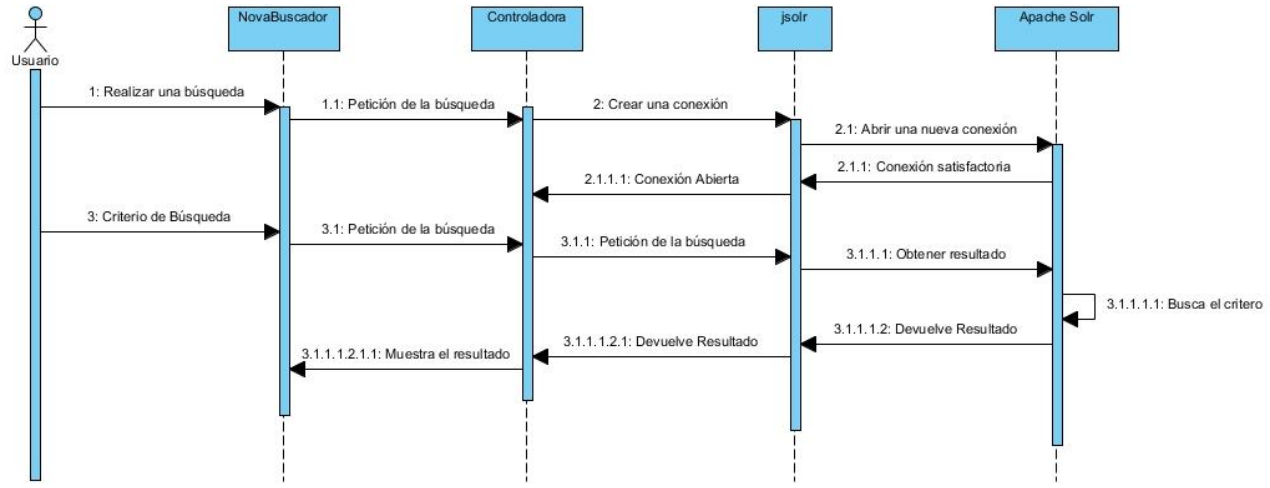


Figura 9: Diagrama de Secuencia Realizar búsqueda.

2.6 Conclusiones parciales

En el presente capítulo se define el buscador para la Distribución Cubana GNU/Linux Nova Escritorio 5.0 con funcionalidades del mismo, para un total de 11 requisitos funcionales y 2 no funcionales. Con estos se obtienen las funcionalidades de configurar conexión, abrir conexión, cerrar conexión, realizar búsqueda, filtrar búsqueda, abrir fichero y acceder a carpetas, para una mejor comprensión de la propuesta actual. Se definen los patrones diseño GRASP experto, creador y controlador; entre los patrones Gof el singleton. Su construcción estará basada en una arquitectura N Capas con un estilo arquitectónico de 3 capas.

Capítulo 3. Implementación y Pruebas de la Solución Propuesta

3.1 Introducción

Durante esta etapa se desarrollan las tareas de programación definidas para darle cumplimiento a la solución propuesta, además se define el estándar de codificación a utilizar en la implementación para la organización del código, así como los componentes y relaciones a tener en cuenta. Durante la etapa de pruebas se persigue encontrar errores cometidos al realizar el diseño y construcción de la solución implementada.

3.2 Implementación de la solución propuesta

Tras analizar y diseñar la solución, corresponde enfrentar la etapa de implementar la aplicación. Donde se elaboran, adaptan y añaden los elementos previamente contemplados. El modelo diseñado en la fase anterior es la guía maestra para comenzar a ejecutar los componentes y programar las funciones con las que se debe hacer la implementación de la aplicación.

Diagramas de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes o bibliotecas cargadas dinámicamente (Sommerville, 2005).

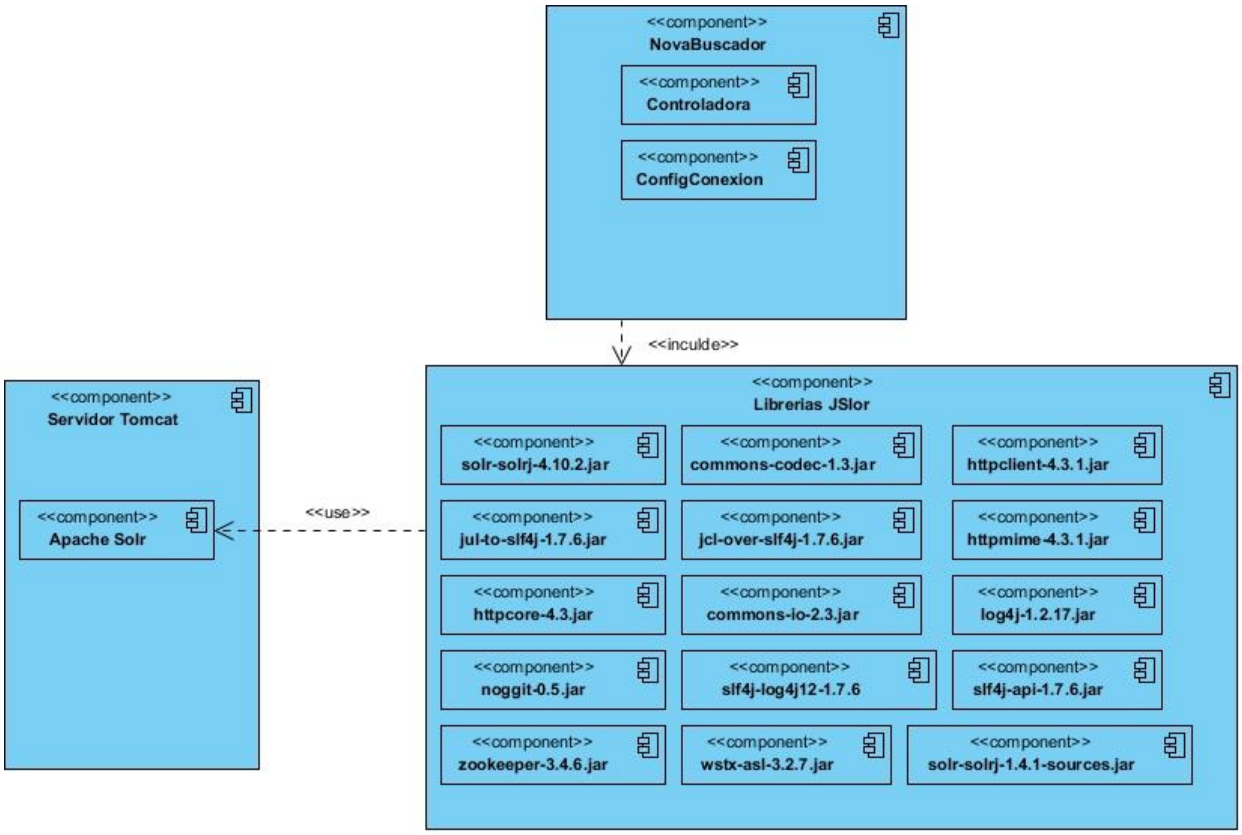


Figura 10: Diagrama de componentes del sistema.

La Figura 10 muestra el diagrama de componentes correspondiente al sistema, en el cual el componente NovaBuscador está compuesto por los componentes Controladora y ConfigConexion, encargados del manejo del sistema, los que, a su vez incluyen el conjunto de librerías JSolr, que contiene los componentes necesarios para la interacción con el servidor Apache Solr encontrado en el servidor Tomcat y realizar las peticiones.

3.3 Estándares de codificación

Un estándar es un modelo, norma, patrón, referencia o la especificación que regula la realización de ciertos procesos o la fabricación de componentes para garantizar la interoperabilidad. En la implementación de la aplicación a desarrollar, se utilizan varios estándares de codificación, que certifican la legibilidad y organización del código de la misma, simplificando esfuerzos a la hora de darle mantenimiento y seguimiento a la aplicación.

A continuación se especifican los estándares de codificación que se utilizan en la construcción de la solución:

Lower Camel Case: similar al *Camel Case,* solo que la primera letra de la primera palabra es también en minúscula (Lago, 2008).

Upper Camel Case: similar al *Camel Case,* siendo la primera letra de la primera palabra en mayúscula (Lago, 2008).

El nombre de las clases debe ser declarado utilizando el estilo *Upper Camel Case* y cuando en un fichero se define el espacio de nombre (*namespace*) debe existir una línea en blanco a continuación de la declaración. Además, cuando están presentes todas las declaraciones de uso, se escriben a continuación de la definición del *namespace* y se debe utilizar la palabra reservada *use* una vez por cada declaración; y además debe concluir con una línea en blanco.

```
 * @author yoandri
 */
public class Conectar {

    private String Url;
    private Collection<SolrInputDocument> docs;
    SolrServer servidor;
```

Figura 11: Estándar de codificación Upper Camel Case.

El nombre de las funciones debe ser declarado utilizando el estilo *Lower Camel Case*.

```
public void indexFilesSolrCell(String fileName, String solrId)
  throws IOException, SolrServerException {


ContentStreamUpdateRequest up = new ContentStreamUpdateRequest("/update/extract");
up.addFile(new File(fileName), fileName);
```

Figura 12: Estándar de codificación Lower Camel Case.

3.4 Pruebas de software

Las pruebas son un conjunto de actividades que se planean con anticipación y se realizan de manera sistemática. Por tanto, se debe definir una plantilla para las pruebas de software (un conjunto de pasos en que se puedan incluir técnicas y métodos específicos del diseño de casos de pruebas) (Pressman, 2005).

Método de Prueba

Como método de prueba se escoge el método de caja negra, con el que se prueban los casos de prueba seleccionados y posibilita comprobar el comportamiento del sistema a través de los requisitos. No se decide aplicar pruebas de caja blanca, dada la complejidad de las funcionalidades, ya que el correcto funcionamiento de ellas depende en muchos de los casos de un conjunto de métodos externos a la aplicación.

Con este método se pretende demostrar que las funciones del software son operativas, la entrada se acepta de forma correcta y se produce una salida correcta.

Con las mismas se pretende encontrar estos tipos de errores:

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

- Errores en estructuras de datos.

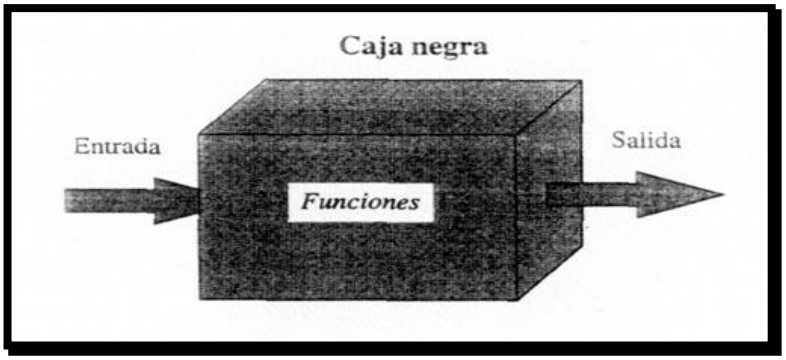


Figura 13: Método de pruebas de Caja negra.

Para la misma se usa la técnica de partición equivalente donde se divide el dominio de entrada de un programa en clases de datos, a partir de las cuales se derivan los casos de prueba. Cada una de estas clases de equivalencia representa un conjunto de estados válidos o inválidos para las condiciones de entrada, la cual se constatará en las pruebas funcionales.

Tipos de prueba

Pruebas funcionales

Las pruebas funcionales están basadas en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. Las mismas se hacen mediante el diseño de modelos de prueba que buscan evaluar cada una de las opciones con las que cuenta el paquete informático. Son pruebas específicas, concretas y exhaustivas para comprobar y validar que el software hace lo que debe y sobre todo, lo que se especificado (Adame, 2009).

Se ejecuta la prueba de este tipo para el caso de uso Configurar Conexión con diferentes entradas, con el objetivo de determinar que los resultados obtenidos fueran los esperados bajo cualquier situación, y así, dar por cumplidos los requerimientos.

Diseño de Casos de Prueba basado en CU

Caso de Prueba "Configurar Conexión"

Escenario	**Descripción**	**V1 dirección**	**Respuesta del sistema**	Flujo Central
EC 1.1 Configurar Conexión con una dirección válida.	La herramienta configura la conexión con el motor de búsqueda.	http://localhost:8080/solr	El sistema configura la conexión con el motor de búsqueda y crea una nueva conexión.	1. El usuario presiona el botón del menú opciones Configurar conexión. 2. El usuario escribe la dirección del servidor. 3. El usuario presiona el botón Guardar.
EC 1.2 Configurar Conexión con una dirección inválida.	La herramienta no configura la conexión con el motor de búsqueda y muestra un mensaje de error con el texto "La dirección es incorrecta. No se pudo conectar".	http://localhost:8080/	El sistema no configura la conexión con el motor de búsqueda y se cierra la conexión.	
EC 1.3 Configurar Conexión dejando el campo dirección vacío.	La herramienta no configura la conexión con el motor de búsqueda y muestra un mensaje de error con el texto "No se ha introducido una dirección".		El sistema no configura la nueva conexión y mantiene la configuración existente.	

3.4.1 Resultados obtenidos de las pruebas realizadas al sistema

Se utilizó el método de caja negra para realizar las pruebas sobre el sistema, donde se encontraron 2 no conformidades, en 2 iteraciones. Una de estas no conformidades se produjo debido a una validación

incorrecta, la cual fue posteriormente resuelta. La otra no conformidad está sujeta a la falta de permisos para acceder a las ubicaciones de archivos en el sistema, la cual no pudo ser resuelta ya que rebasa el alcance de la investigación; pues requiere permisos de administración tanto para la ejecución de la aplicación como para el motor de búsqueda de indexación que estará desplegado en el servidor *Web*.

En la siguiente gráfica se muestran las iteraciones realizadas, y las no conformidades detectadas.

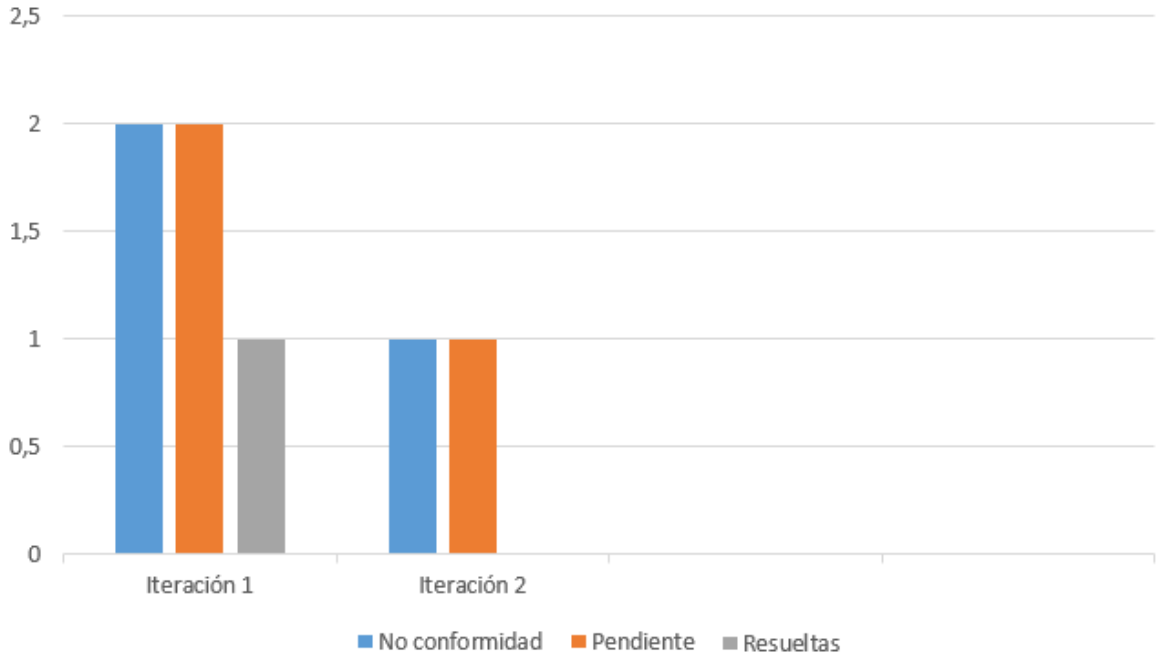


Figura 14: Resultado de las pruebas funcionales.

Se realizaron pruebas de funcionales donde se pudo constatar que el sistema cumple puntualmente con todos los requisitos planteados. Al término de las pruebas se obtuvo un producto con calidad que cumple con todas las especificaciones inicialmente planteadas.

3.5 Aportes de la investigación

El desarrollo de la solución les brindará a los usuarios de la Distribución Cubana de GNU/Linux Nova Escritorio 5.0, un producto que les permitirá realizar búsquedas más profundas en el sistema basadas en la indexación y en el análisis de los metadatos de los mismos y contribuirá a lograr la soberanía tecnología a la que avanza el país.

3.6 Conclusiones parciales

En presente el capítulo se muestran los componentes que van a conformar la propuesta de solución, así como los estándares Upper Camel Case y Lower Camel Case a utilizar en la implementación para mejor estructuración del código. También se describen las pruebas realizadas a la aplicación para comprobar su correcto funcionamiento, en las cuales se obtuvieron un total de 2 conformidades.

Conclusiones

El presente trabajo da cumplimiento a las tareas trazadas, enfatizando de manera general se llega a las siguientes conclusiones:

1. El estudio de las soluciones homólogas permitió definir las características y cualidades de la solución propuesta.
2. Se obtuvo un sistema que permite realizar búsquedas básicas y basadas en índices; donde se analiza el conjunto de los archivos del sistema, su gestión y filtrado por diferentes parámetros definidos.
3. Se realizaron pruebas exitosas con las que se comprobó el correcto funcionamiento del sistema.

Recomendaciones

Al término de esta investigación se recomienda:

1. El estudio de formas de control y acceso para posibilitar el acceso pleno a todo el contenido del sistema.

2. Incluir una funcionalidad de reconocimiento de texto en imágenes, con el objetivo de seguir enriqueciendo los resultados a obtener en las búsquedas.

Referencias Bibliográficas

Albert, Mª Isabel Espí, y otros. 2002. **[En línea] 2002.**

Alegsa, Leandro. 2013. **DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA.** ***Definición de Archivo Informático.*** **[En línea] 2013. http://www.alegsa.com.ar/Diccionario/diccionario.php.**

Alvarez, Luis Alvarez. 2009. **[En línea] 2009. e-archivo.uc3m.es/bitstream/handle/.../PFC_Luis_Alvarez_Alvarez.pdf?.**

2014. **Apache Solr. [En línea] 2014. http://lucene.apache.org/solr/.**

Castillo, Genesis Cristel Lara. 2015. **Prezi. [En línea] 2015. https://prezi.com/oqrwf5j4d5pl/motores-de-busqueda/.**

2013. **CEC. [En línea] 2013. http://www.cec.uchile.cl/~luvasque/edo/java/manuales/JVM%20y%20variables%20de%20entorno.pdf.**

Ceria, Santiago. 2001. **[En línea] 2001. http://www-2.dc.uba.ar/materias/isoft1/2001_2/apuntes/CasosDeUso.pdf.**

2015. **concepto.de. [En línea] 2015. http://concepto.de/cpu/.**

Corrales, Juan Desongles. 2005. ***Ayudantes Técnicos de Informática.*** **2005.**

Cuevas, Laura Nathalia Uribe. 2014. ***Metabuscadores.*** **2014.**

Echevarría, Jorge Ignacio Sánchez. 2008. ***Elaboración de un Prototipo de Buscador de la Facultad de Ciencias.*** **2008.**

Franganillo, Jorge y Figuerola, Teresa Maria. 2001. **Bid. [En línea] 2001. http://bid.ub.edu/06frang2.htm.**

Gavilán, César Martín. 2009. ***Concepto y función de archivo.*** **2009.**

Gil, Dr.Ignacio. 2004. **[En línea] 2004. http://personales.upv.es/igil/java.PDF.**

Hall, Prentice. 2003 . **Lisis. [En línea] 2003 . http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf.**

Referencias bibliográficas

Hernández, Yanirys Montes de Oca y Díaz, Yuliesky Brito. 2009. ***LA GESTIÓN DE INFORMACIÓN DE TRÁMITES PROTOCOLIZABLES COMPLEJOS.*** **2009.**

Juarez, Efrain Alberto Olguin. 2013. **Metodología Open UP. [En línea] 2013. http://openupeaojmp.blogspot.com/2013/09/metodologia-open-up.html.**

2015. **Kioskea. [En línea] 2015. http://es.kioskea.net/contents/275-protocolos.**

Kuć, Rafał. 2013. ***Apache Solr 4 Cookbook.*** **s.l. : Packt Publishing, 2013.**

Lago, Diego. 2008. ***Guía de Estilo de Progrmación.*** **2008.**

Laguillo, Luis Fernando Romero. 1998. ***Publicar en Internet: guía practica para la creación de documentos html.*** **1998.**

Larman, Carig. 1999. ***UML y Patrones.*** **s.l. : PRENTICE HALL, Mexico, 1999.**

Latorre, Guillermo. 2008. **Genbeta. [En línea] 15 de mayo de 2008. http://www.genbeta.com/buscadores/tracker-mucho-mas-que-un-buscador.**

Lerman, Carig. 2003. ***UML y Patrones.*** **2003.**

Luna, Katy de la. 2012. **Scridb. [En línea] 2012. http://es.scribd.com/doc/64176230/Un-archivo-o-fichero-informatico-es-un-conjunto-de-bits-almacenado-en-un-dispositivo#scribd.**

Marco, Bartolomé Sintes. 2014. **[En línea] 2014. http://www.mclibre.org/consultar/xml/lecciones/xml_quees.html.**

Mayer, Miguel Ángel y Leisa, Ángela. 2010. **SienceDirect. [En línea] 2010. http://www.sciencedirect.com/science/article/pii/S0212656709005083.**

Méndez, Eva y Senso, José A. 2004. **SEDIC. [En línea] 2004. http://www.sedic.es/autoformacion/metadatos/tema1.htm.**

Morón, y otros. 2004. ***NUEVAS TENDENCIAS DE LA CONTADURÍA PÚBLICA.*** **2004.**

2015. **NetBeans. [En línea] 2015. https://netbeans.org/index_es.html.**

Ochando, Manuel Blázquez. 2013. ***Técnicas avanzadas de recuperación de información.*** **2013.**

Oliván, José Antonio Salvador y Avilés, Rosario Arquero. 2006. ***Una aproximación al concepto de recuperación de infromación en el marco de la ciencia de la documentación.*** **2006.**

Referencias bibliográficas

Pablo. 2009. **Geeks & Linux Atelier. [En línea] 2009. http://glatelier.org/2009/04/18/gnome-do-instalacion-y-extensiones-utiles/.**

Palma, Wencesalo. 2005. ***Almacenamiento y Recuperación de Información.* 2005.**

Perez, J. 2005. ***Metadatos.* 2005.**

2014. **Picodotdev. [En línea] 2014. http://picodotdev.github.io/blog-bitix/2014/04/introduccion-a-elasticsearch/.**

Pressman, Roger. 2005. ***Ingeneria de Software un Enfoque práctico.* 2005.**

Quiroga, Juan Pablo. 2014. **[En línea] 2014. https://sistemas.uniandes.edu.co/~csof5101/dokuwiki/lib/exe/fetch.php?media=principal:csof5101-requerimientos.pdf.**

Sommerville. 2005. ***Ingenería de Software.* 2005.**

Suela, Eduardo. 2013. **Prezi. [En línea] 2013. https://prezi.com/ozbhwnbcvneb/copy-of-plataformas-de-desarrollo-frameworks-comerciales/.**

2009. **thefreedictionary. [En línea] 2009. http://es.thefreedictionary.com/indexaci%C3%B3n.**

Tolosa, Gabriel H. y Bordignon, Fernando R.A. 2005. **[En línea] 2005. http://eprints.rclis.org/12243/1/Introduccion-RI-v9f.pdf.**

Velasco, Marta Godoy. 2005. **LA INDIZACIÓN EN LA DOCUMENTACIÓN. [En línea] 2005. http://www.galeon.com/indizacion/indizacion.html.**

Anexo 1. Descripción de CU

CU 2. Gestionar Ficheros.

Objetivo	Gestionar Ficheros.
Actores	Usuario
Resumen	Posibilita acceder a los diferentes directorios del sistema así como abrir carpetas y archivos del sistema.
Complejidad	Alta
Prioridad	Alta
Precondiciones	
Postcondiciones	

Flujo de eventos		
	Actor	**Sistema**
1.	Hace doble clic en una carpeta.	Borra la lista de ficheros y carpetas y muestra una nueva lista de ficheros y carpetas con los ficheros y carpetas que se encuentran dentro de esta.
2.	Hace doble clic en un fichero.	Se mantiene en el directorio actual y se ejecuta el fichero seleccionado.

3.	Hace clic en una carpeta.	Muestra los valores de los metadatos de la carpeta seleccionada.
4.	Hace clic en un fichero.	Muestra los valores de los metadatos del archivo seleccionado.
5.		Termina el Caso de Uso.
Flujos alternos		
1 No se puede acceder a la carpeta.		
	Actor	**Sistema**
1.	Hace doble clic en una carpeta.	El sistema muestra un mensaje de error.
Flujos alternos		
2 No se puede acceder al archivo.		
	Actor	**Sistema**
1.	Hace doble clic en un archivo.	El sistema muestra un mensaje de error que no se puede acceder al archivo con la dirección especificada.

CU 3. Realizar Búsquedas.

Objetivo	Realizar búsqueda.

Actores	Usuario
Resumen	Posibilita realizar una búsqueda en el sistema.
Complejidad	Muy Alta
Prioridad	Crítico
Precondiciones	
Postcondiciones	

Flujo de eventos		
	Actor	**Sistema**
1.	Introduce un criterio de búsqueda. Hace clic en el botón “Buscar”	El sistema realiza la búsqueda por nombre de todo aquel fichero que en su nombre contenga el criterio de búsqueda y devuelve una lista de estos.
2.	Introduce un criterio de búsqueda. Hace clic en el botón “Búsqueda Avanzada”	El sistema realiza una búsqueda mediante el uso de indexación en los ficheros que se encuentran indexados en el sistema y devuelve una lista con los mismos.
3.		Termina el Caso de Uso.

Flujos alternos
1 No se introduce un criterio de búsqueda.

	Actor	Sistema
1.	Hace clic en el botón “Buscar”	El sistema devuelve una lista con todos los ficheros contenidos en el directorio y sus subdirectorios.
2.	Hace clic en el botón “Búsqueda Avanzada”	El sistema devuelve una lista con los primeros 500 ficheros encontrados.
3.		Termina el Caso de Uso.

CU 4. Gestionar Filtros de Búsqueda.

Objetivo	Gestionar Filtros de Búsqueda.	
Actores	Usuario	
Resumen	Posibilita realizar búsquedas por un metadato específico definido en el sistema.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones		
Postcondiciones		
Flujo de eventos		
	Actor	**Sistema**

1.	Hace clic en filtros.	Despliega la lista de filtros.
2.	Selecciona un criterio para filtrar. Hace clic en el botón Búsqueda Avanzada.	El sistema realiza una búsqueda mediante indexación solo de aquellos documentos que cumplan con el criterio de búsqueda, el filtro seleccionado y devuelve una lista con los mismos.
3.		Termina el caso de uso.
Flujos alternos		
1 No se selecciona un filtro.		
	Actor	Sistema
1.	Hace un clic en el botón Búsqueda Avanzada.	El sistema realiza una búsqueda por nombre y devuelve una lista con los mismos.
2.		Termina el caso de uso.

CU 5. Abrir Conexión

Objetivo	Abrir Conexión.
Actores	Usuario

Resumen	Posibilita comprobar o abrir una conexión con el motor de búsqueda en caso de que se encuentre cerrada.
Complejidad	Media
Prioridad	Media
Precondiciones	
Postcondiciones	

Flujo de eventos

	Actor	**Sistema**
1.	Hace clic en el menú de opciones en la opción Configurar Conexión y dentro de esta selecciona Abrir Conexión.	El sistema comprueba si la conexión se encuentra establecida, sino crea una nueva y finalmente muestra un mensaje donde aparecen los datos del servidor. (Si ocurre un error, sucede el flujo alterno 1)
2.		Termina el Caso de Uso.

Flujos alternos

1 Error en la conexión.

	Actor	**Sistema**
1.		No se crea la conexión.

2.		Muestra un mensaje de error al conectar.

CU 6. Cerrar Conexión.

Objetivo	Cerrar Conexión.	
Actores	Usuario	
Resumen	Posibilita finalizar una conexión con el motor de búsqueda.	
Complejidad	Media	
Prioridad	Media	
Precondiciones		
Postcondiciones		
Flujo de eventos		
	Actor	**Sistema**
1.	Hace clic en el menú de opciones en la opción Configurar Conexión y dentro de esta selecciona Cerrar Conexión.	El sistema comprueba si la conexión se encuentra establecida, termina esta y finalmente muestra un mensaje de fin de conexión (Si ocurre error, sucede el flujo alterno 1)
2.		Termina el Caso de Uso.
Flujos alternos		

1 Error en la conexión.		
	Actor	**Sistema**
1.		No se cierra la conexión.
2.		Muestra mensaje de error al finalizar la conexión.

Anexo 2. Casos de Prueba

Caso de Prueba. CU 2. Gestionar archivos

Escenario	Descripción	V1 Dirección de ficheros	Respuesta del sistema	Flujo Central
EC 1.1 Acceder a las carpetas del sistema	La herramienta muestra una lista con los ficheros y carpetas de la PC	/	El sistema muestra una lista con los ficheros y carpetas	1) El usuario presiona doble clic sobre el fichero o la carpeta
EC 1.2 Acceder a las carpetas del sistema	La herramienta no muestra una lista con los ficheros y carpetas de la PC	/etc	El sistema muestra una lista vacía	
EC 1.3 Ejecutar un fichero	La herramienta manda al sistema a ejecutar el archivo.	/home/Yoandri/Escritorio/Tesis.doc	El sistema ejecuta el fichero seleccionado.	
EC 1.4 Ejecutar un fichero	La herramienta no muestra una lista con los ficheros y carpetas de la PC	/etc/cntlm.config	El sistema muestra un mensaje de error de acceso al fichero cntlm.config	

Caso de Prueba. CU 3. Realizar búsquedas

Escenario	**Descripción**	**V1 buscar**	**Respuesta del sistema**	Flujo Central
EC 1.1 Obtener resultados de la búsqueda con indexación	La herramienta realiza la búsqueda en los ficheros indexados en el motor de búsqueda	java	El sistema muestra una lista con los ficheros encontrados	1) El usuario presiona el botón del menú opciones Conectar con el Servidor 2) El usuario presiona el botón Realizar
EC 1.2 Obtener resultados de la búsqueda con indexación	La herramienta realiza la búsqueda en los ficheros indexados en el	moda	El sistema muestra una lista vacía y notifica que no se encontró ningún	

	motor de búsqueda y no encuentra resultado		criterio de búsqueda	búsque-das.

Caso de Prueba. CU 4. Gestionar Filtros.

Escenario	**Descripción**	**V1 buscar**	**Respuesta del sistema**	Flujo Central
EC 1.1 Obtener resultados de la búsqueda con indexación filtrada por nombre.	La herramienta realiza la búsqueda en los ficheros indexados en el motor de búsqueda filtrando solo por el metadato del nombre.	java	El sistema muestra una lista con los ficheros que en el nombre contienen la palabra java.	1) El usuario despliega la lista de filtros se-lecciona un filtro. 2) El usuario presiona el botón Realizar búsqueda avan-zada.
EC 1.2 Obtener resultados de la búsqueda con indexación filtrada por Autor.	La herramienta realiza la búsqueda en los ficheros indexados en el motor de búsqueda filtrando solo por el metadato del Autor.	Pressman	El sistema muestra una lista con los ficheros que en su lista de autores contienen el nombre Pressman.	
EC 1.3 Obtener resultados de la búsqueda con indexación filtrada por Contenido.	La herramienta realiza la búsqueda en los ficheros indexados en el motor de búsqueda donde analiza el contenido de los ficheros.	Indexación	El sistema muestra una lista con los ficheros que en su contenido contienen la palabra Indexación.	
EC 1.4 Obtener resultados de la búsqueda con	La herramienta realiza la búsqueda en los	21/04/2015	El sistema muestra una lista con los ficheros	

indexación filtrada por Fecha de Modificación.	ficheros indexados en el motor de búsqueda donde analiza su última fecha de modificación o creación.		que en su fecha de modificación o creación contengan la fecha.	

Anexo 3. Diagramas de Secuencia

Diagrama de Secuencia. CU 1. Configurar Conexión.

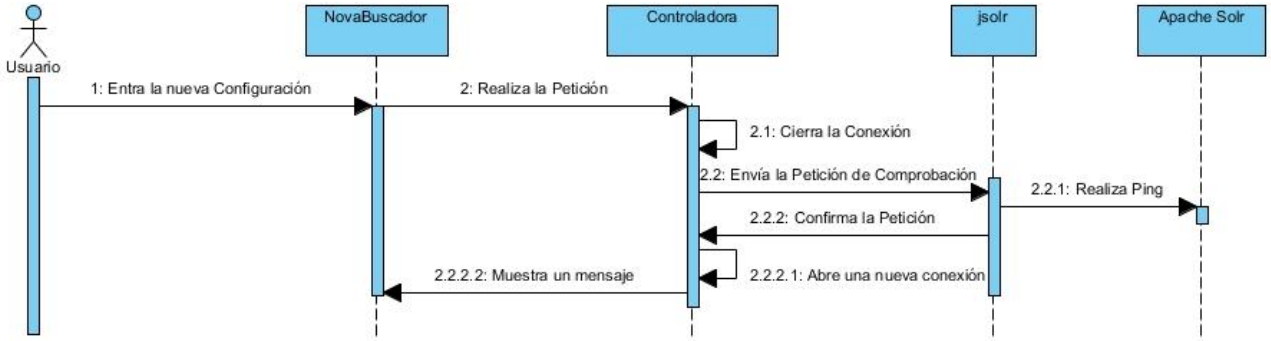


Diagrama de Secuencia. CU 2. Gestionar Ficheros.

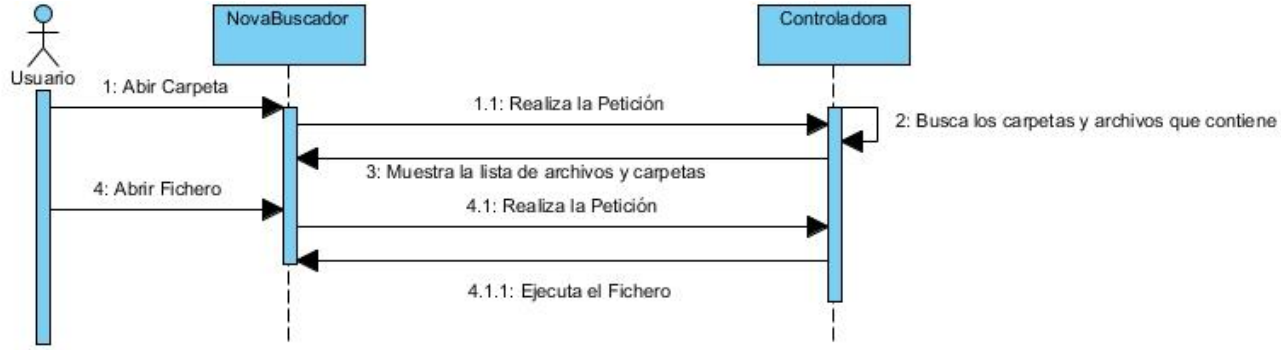


Diagrama de Secuencia. CU 4. Gestionar Filtros.

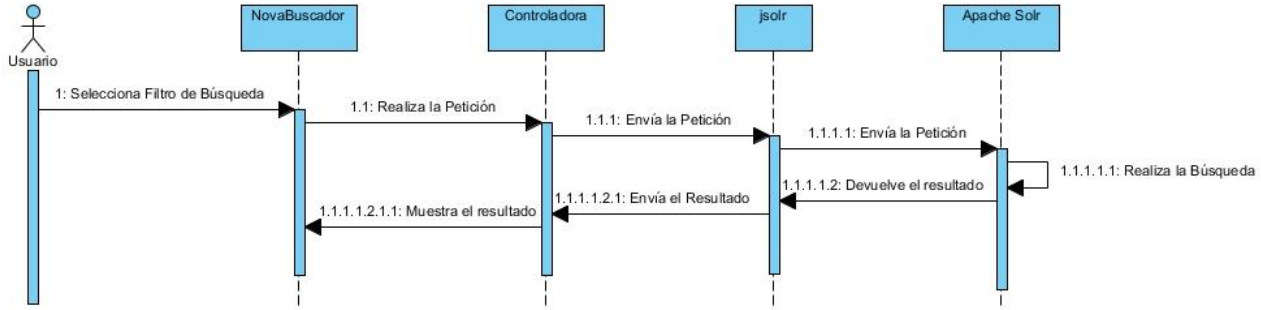


Diagrama de Secuencia. CU 5. Abrir Conexión.

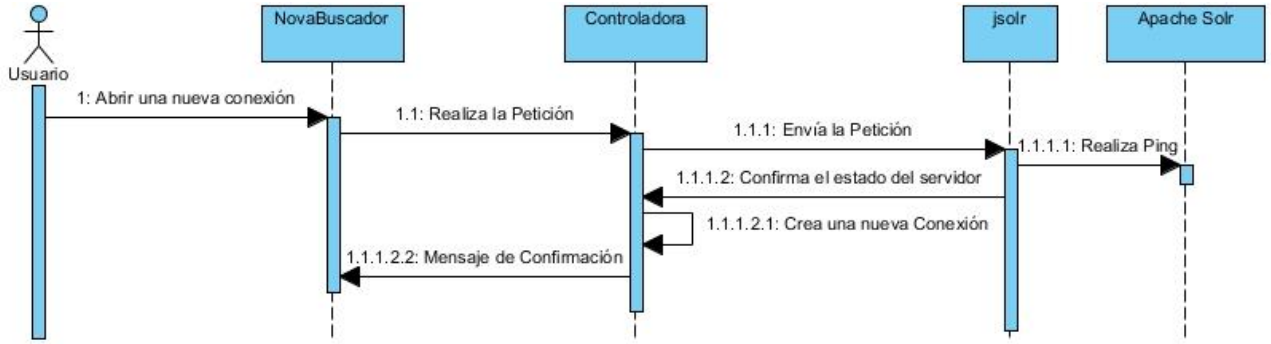
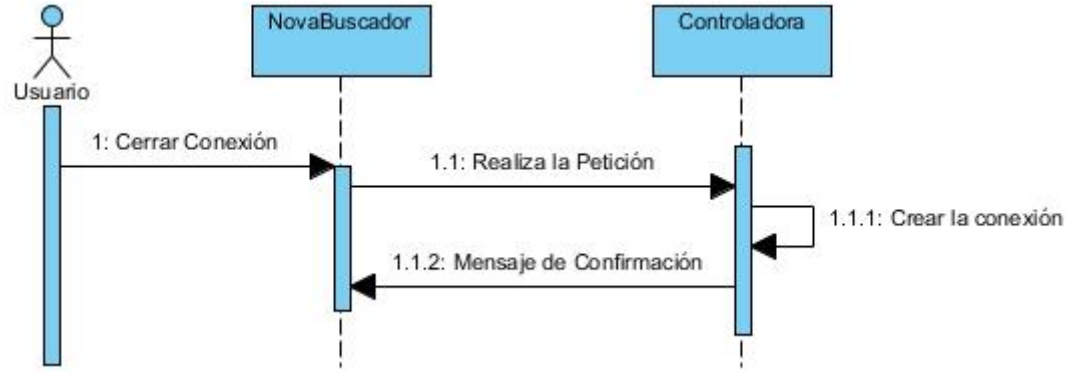


Diagrama de Secuencia. CU 6. Cerrar Conexión.



Glosario de términos

Archivo: Un archivo es uno o más documentos, sea cual sea su fecha, su forma y soporte material, acumulados en un proceso natural por una persona o institución pública o privada en el transcurso de su gestión, conservados, respetando aquel orden, para servir como testimonio e información para la persona o institución que lo produce, para los ciudadanos o para servir de fuentes de historia (Gavilán, 2009).

Índice: Valor que hace referencia a la posición de un elemento dentro de un arreglo. Los índices son valores numéricos naturales (Corrales, 2005).

Índice primario: Es construido sobre un campo que a su vez se utiliza para ordenar el archivo de datos (Palma, 2005).

Índice secundario: Se construye sobre un campo de ordenamiento que abarca varios registros con el mismo valor, dentro de un archivo ordenado de registros (Palma, 2005).

Indización: Técnica del análisis documental para representar y describir el contenido de los documentos, mediante conceptos principales contenidos en ellos (palabras clave) o vocabularios controlados (descriptores, términos o encabezamientos de materia), con el fin de guiar al usuario en la recuperación de los documentos que necesita (Velasco, 2005).

Metabuscador: Es un buscador de buscadores. Una potente herramienta que realiza rastreos por diferentes bases de datos, proporcionando una combinación de los mejores resultados. Comúnmente se les denomina robots, arañas o gusanos (Cuevas, 2014).

CPU: Esta denominación es una abreviación que refiere a una Unidad Central de Procesamiento. Es un componente básico de la computadora personal u ordenador que procesa datos y realiza cálculos matemá-ticos-informáticos. Proporciona la capacidad de programación, y junto con la memoria y los dispositivos de entrada/salida es de los componentes computacionales que encontramos presente en toda la historia de las computadoras (concepto.de, 2015).